Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Πανεπιστήμιο Θεσσαλίας

Ανάπτυξη σημείων καταχώρησης σε πολυεπίπεδα δίκτυα

Deployment of check-in nodes in complex networks

Διπλωματική εργασία του Σαμαρά Λεωνίδα

Επιβλέποντες καθηγητές:

Κατσαρός Δημήτριος, Επίκουρος Καθηγητής

Μποζάνης Παναγιώτης, Καθηγητής

Ευχαριστίες

Με την περάτωση της παρούσας εργασίας, θα ήθελα να ευχαριστήσω θερμά τους επιβλέποντες καθηγητές μου, κ. Κατσαρό Δημήτριο και κ. Μποζάνη Παναγιώτη, για την δυνατότητα που μου έδωσαν να εργαστώ πάνω σε ένα τόσο ενδιαφέρον θέμα. Επίσης, οφείλω ένα μεγάλο ευχαριστώ στο Μπασάρα Παύλο για την αμέριστη βοήθεια που μου παρείχε σε όλη τη διάρκεια της εργασίας.

Τέλος, είμαι ιδιαίτερα ευγνώμων προς την οικογένεια και τους φίλους μου για την απεριόριστη στήριξη που μου δείξαν όλα αυτά τα χρόνια. Χωρίς αυτούς τίποτα δεν θα ήταν εφικτό.

Abstract

In many real complex networks such as the city road networks and highway networks, vehicles often have to pass through some specially functioned nodes to receive check-in like services such as gas supplement at gas stations. The location selection of all check-in nodes is very essential and important and in order to solve this problem we define only one condition, every shortest path includes at least one check-in node. In this work, we simulate scale free and random networks and then we use our algorithm in real networks. The deployment of check-in nodes still remains an open problem in complex network studies.

Σύνοψη

Σε πολλά πραγματικά σύνθετα δίκτυα υπάρχουν κάποιοι κόμβοι που παρέχουν συγκεκριμένες υπηρεσίες στο δίκτυο. Αυτούς τους κόμβους τους ονομάζουμε κόμβους καταχώρησης. Σε ένα οδικό δίκτυο τα οχήματα θα πρέπει να ανεφοδιαστούν με καύσιμα. Σε ένα δίκτυο ηλεκτροδότησης ή ένα δίκτυο τηλεπικοινωνιών, αν ένα μέρος των κόμβων υποστεί κάποια βλάβη, μπορεί να καταρρεύσει ολόκληρο το δίκτυο. Με την προϋπόθεση πως κάθε συντομότερο μονοπάτι περιέχει τουλάχιστον έναν κόμβο καταχώρησης , η επιλογή των κόμβων καταχώρησης αποτελεί μια εξαιρετικά ουσιώδη και σημαντική διαδικασία. Στο παρόν έγγραφο θα χρησιμοποιήσουμε τις ιδιότητες των σύνθετων δικτύων για να αναγνωρίσουμε αυτούς τους κόμβους. Θα προσομοιώσουμε διάφορα είδη δικτύων και θα συγκρίνουμε τα αποτελέσματα μεταξύ τους.

Περιεχόμενα

Κεφάλαιο 1: Περιγραφή προβλήματος

Κεφάλαιο 2: Σύνθετα δίκτυα και ιδιότητες

2.1: Τύποι σύνθετων δικτύων

2.1.1: Άνευ κλίμακας (scale free)

2.1.2: Τυχαία δίκτυα (random networks)

2.2: Κεντρικότητες σύνθετων δικτύων

2.2.1: Degree centrality

2.2.2: Betweenness centrality

2.2.3: Power community index (PCI)

2.2.4: Closeness centrality

2.2.5: K Shell

Κεφάλαιο 3: Αλγόριθμος επίλυσης

3.1: Ψευδοκώδικας

3.2: Υλοποίηση

Κεφάλαιο 4: Προσομοιώσεις

4.1: Άνευ κλίμακας δίκτυα (Scale free, Barabási–Albert μοντέλο)

4.1.1: Κατευθυνόμενα

4.1.2: Μη κατευθυνόμενα

4.2: Τυχαία δίκτυα Μοντέλο Erdős–Rényi

4.2.1: Κατευθυνόμενα

4.2.2: Μη κατευθυνόμενα

4.3: Σύγκριση κεντρικοτήτων για τους τύπους δικτύων

4.3.1: Κατευθυνόμενα δίκτυα

4.3.2: Μη κατευθυνόμενα δίκτυα

Κεφάλαιο 5: Εφαρμογή αλγορίθμου σε πραγματικά δίκτυα

Κεφάλαιο 6: Μελλοντική εργασία

Κεφάλαιο 7: Βιβλιογραφία

Παράρτημα Α

Κατευθυνόμενα δίκτυα

Μη κατευθυνόμενα δίκτυα

Κεφάλαιο 1

Εισαγωγή

Καθημερινά βασιζόμαστε σε διαφόρων ειδών τεχνητά δίκτυα, μερικά από αυτά είναι τα οδικά δίκτυα, τα δίκτυα ηλεκτροδότησης, τα δίκτυα τηλεπικοινωνιών, τα κοινωνικά δίκτυα το ίντερνετ αλλά και πολλά άλλα. Η επιστήμη των σύνθετων δικτύων μπορεί να μοντελοποιήσει αυτά τα πραγματικά δίκτυα με μεγάλη επιτυχία και μελετώντας τα χαρακτηριστικά των σύνθετων δικτύων μπορούμε να μελετήσουμε και να βγάλουμε συμπεράσματα για τα πραγματικά μας δίκτυα.

Σε κάθε σύνθετο δίκτυα, ένα μέρος των κόμβων κατέχουν ξεχωριστές λειτουργικότητες παρέχοντας στο δίκτυο ξεχωριστές υπηρεσίες. Αυτοί οι ξεχωριστοί κόμβοι μπορούν να ονομαστούν κόμβοι καταχώρησης και οντότητες που χρησιμοποιούν το δίκτυο πρέπει να περάσουν από τους κόμβους καταχώρησης και να λάβουν ξεχωριστές λειτουργίες. Στο παράδειγμα του οδικού δικτύου, τα οχήματα θα πρέπει να περάσουν από βενζινάδικα (κόμβος καταχώρησης) για να εφοδιαστούν (ξεχωριστή λειτουργεία). Στα δίκτυα ηλεκτροδότησης και τηλεπικοινωνιών, ποιοι και πόσοι κόμβοι καταχώρησης μπορούν να τεθούν εκτός λειτουργείας πριν τεθεί ολόκληρο το δίκτυο χωρίς ηλεκτροδότηση και επικοινωνία (ξεχωριστή λειτουργεία).

Με όσα αναφέρθηκαν παραπάνω, προκύπτει το εξής πρόβλημα. Ποιοι είναι οι κόμβοι καταχώρησης σε ένα σύνθετο δίκτυο/πραγματικό δίκτυο και πως μπορώ να ελαχιστοποιήσω αυτούς τους κόμβους; Για να λύσουμε το παραπάνω πρόβλημα, ορίζουμε μια και μοναδική προϋπόθεση. Κάθε συντομότερο μονοπάτι που υπάρχει στο δίκτυο, θα περιέχει τουλάχιστον έναν κόμβο καταχώρησης. Με αυτή την προϋπόθεση εξασφαλίζουμε τα εξής:

- Οι κόμβοι καταχώρησης θα καλύπτουν ολόκληρο το δίκτυο. Δεν θα υπάρχουν μέρη του δικτύου όπου οι κόμβοι καταχώρησης δεν θα μπορούν να παρέχουν τις υπηρεσίες τους.
- Μπορούμε να χρησιμοποιήσουμε τα χαρακτηριστικά των σύνθετων δικτύων για να υπολογίσουμε πόσα συντομότερα μονοπάτια περνάνε από τον κάθε κόμβο και επομένως πόσο σημαντικός είναι ο κάθε κόμβος

Σε αυτή την εργασία έγιναν προσομοιώσεις σε δίκτυα άνευ κλίμακας (scale free) και τυχαία δίκτυα (random networks). Επίσης οι κεντρικότητες που χρησιμοποιήθηκαν είναι οι degree centrality, betweenness centrality, closeness centrality, Power Community Index (PCI) και k shell. Θα συγκρίνουμε τα αποτελέσματα μεταξύ τους και κάθε είδος δικτύου.

Κεφάλαιο 2

Σύνθετα δίκτυα και ιδιότητες

Αυτό το κεφάλαιο κάνει μια εισαγωγή στα σύνθετα δίκτυα και στις κεντρικότητες αυτών. Αναγνώστες που είναι γνώριμοι με αυτές τις έννοιες, μπορούν να προχωρήσουν στα παρακάτω κεφάλαια.

2.1: Τύποι σύνθετων δικτύων

2.1.1: Άνευ κλίμακας (Scale free)

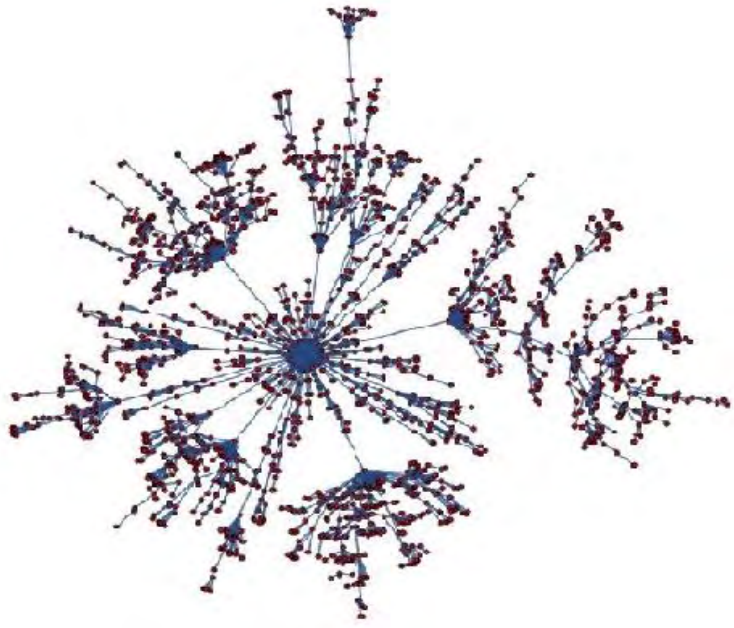
Μια κατηγορία σύνθετων δικτύων είναι τα δίκτυα άνευ κλίμακας (scale free). Στα scale free δίκτυα η κατανομή βαθμών ακολουθεί έναν δυναμονόμο. Αυτό σημαίνει πως θα υπάρχουν μερικοί κόμβοι όπου θα έχουν αρκετά μεγαλύτερο βαθμό από το μέσο βαθμό του δικτύου. Τέτοιοι κόμβοι ονομάζονται και hubs. Τα scale free δίκτυα είναι στενά συνδεδεμένα με τον κανόνα 80 – 20. Το 20% των κόμβων κατέχει το 80% των συνδέσεων του δικτύου, το οποίο είναι λογικό αν σκεφτεί κανείς πως οι hubs κόμβοι συνδέονται με πολλούς κόμβους.

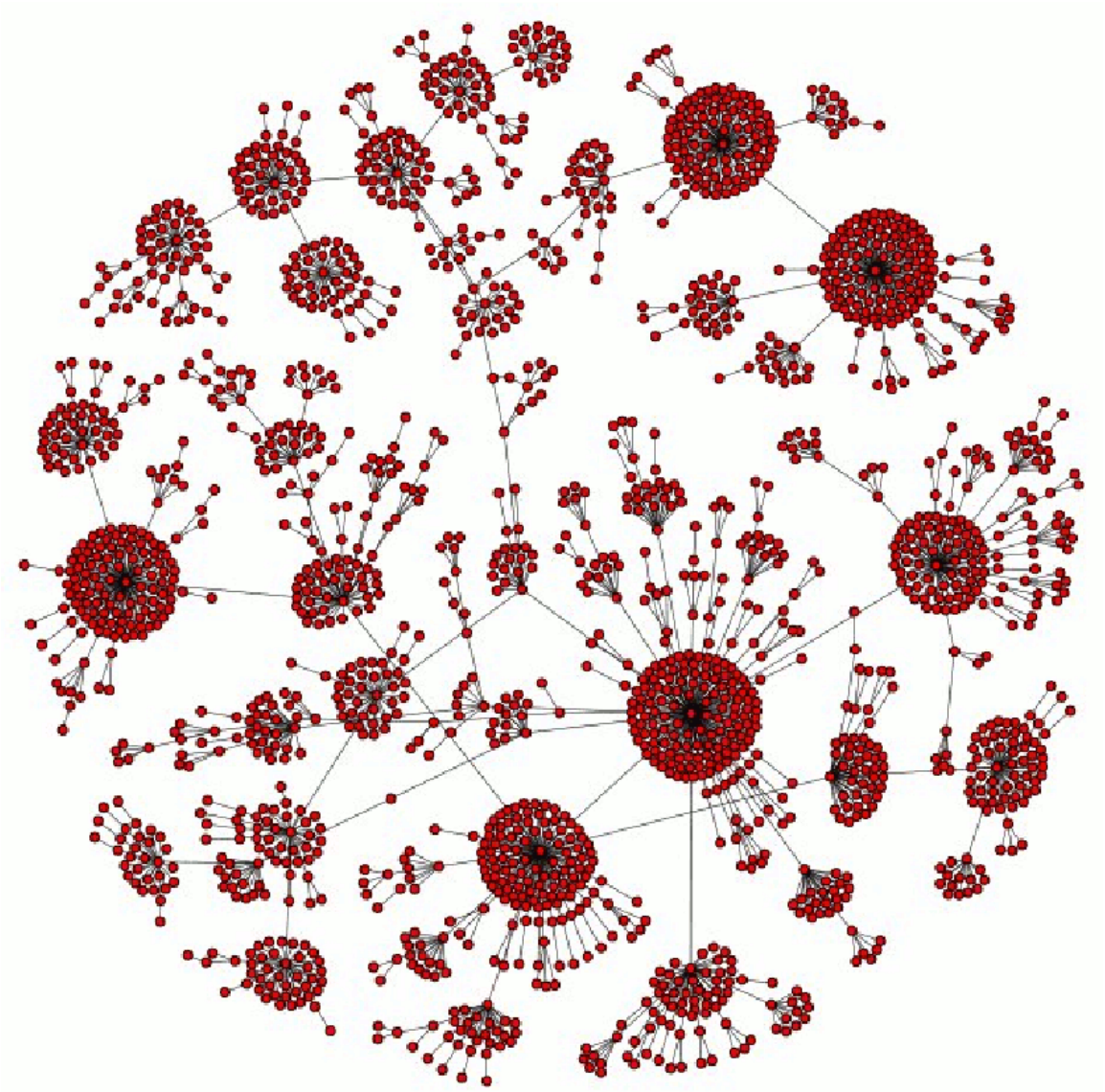
Δύο είναι τα χαρακτηριστικά των scale free δικτύων.

- Συνεχής αύξηση του δικτύου
  Το δίκτυο θα αυξάνεται με την πάροδο του χρόνου.

- Προνομιακή προσκόλληση
  Οι νέοι κόμβοι που θα προστίθενται στο δίκτυο, είναι πιο πιθανό να συνδεθούν με κόμβους του δικτύου που έχουν ήδη πολλές συνδέσεις.

Με άλλα λόγια οι καινούριοι κόμβοι είναι πιο πιθανό να συνδεθούν με τους hubs κόμβους. Οι Barabasi και Albert ήταν οι πρώτοι που πρότειναν ότι τα scale free δίκτυα οφείλονται σε αυτούς τους δυο μηχανισμούς. Οι παρακάτω εικόνες δείχνουν πως μοιάζουν τα scale free δίκτυα. (Παρατηρήστε τους hubs κόμβους)





Πολλά πραγματικά δίκτυα ανήκουν στην κατηγορία των scale free δικτύων και για αυτό θα τα μελετήσουμε. Χαρακτηριστικά παραδείγματα πραγματικών scale free δίκτυών αποτελούν τα τεχνολογικά, κοινωνικά και βιολογικά δίκτυα όπως το internet, το world wide web και οι αλληλεπιδράσεις μεταξύ πρωτεϊνών.

2.1.2: Τυχαία δίκτυα (random networks)

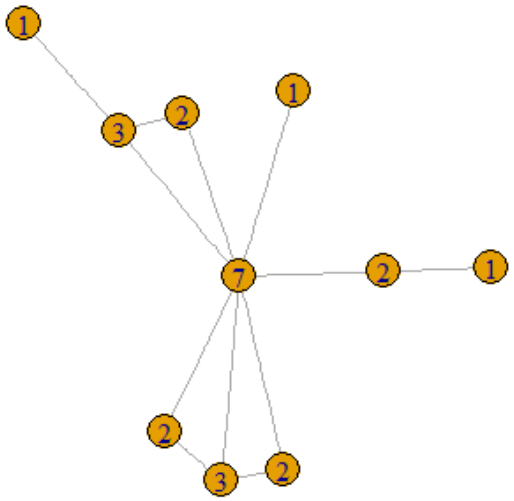
Μια ακόμα κατηγορία των σύνθετων δικτύων αποτελούν και τα τυχαία δίκτυα. Στα τυχαία δίκτυα, με δεδομένο πλήθος κόμβων, το να υπάρχει μια ακμή μεταξύ ένα ζευγάρι κόμβων ορίζεται από μια πιθανότητα p. Στα τυχαία δίκτυα ο κάθε κόμβος μπορεί να είναι συνδεδεμένος με οποιοδήποτε άλλο κόμβο, ακόμα και αν ο ένας κόμβος είναι πολύ μακριά από τον άλλον. Υπάρχουν δύο μοντέλα παραγωγής τυχαίων δικτύων, το μοντέλο του Edgar Gilbert, και το μοντέλο των Erdős–Rényi. Και στα δύο μοντέλα οι κόμβοι που θα υπάρχουν στο δίκτυο είναι δεδομένοι από την αρχή. Στο πρώτο μοντέλο, η πιθανότητα να υπάρχει ακμή μεταξύ δυο κόμβων καθορίζεται από μια πιθανότητα p, ενώ στο δεύτερο μοντέλο και το πλήθος των κόμβων και το πλήθος των ακμών είναι καθορισμένο από τη αρχή. Οι παρακάτω εικόνες δείχνουν πως μοιάζουν τα random δίκτυα. Παρατηρήστε πως αν η πιθανότητα ύπαρξης ακμής είναι πολύ μικρή, τότε θα υπάρχουν κόμβοι που δεν θα συνδέονται με κανέναν. Αντιθέτως, αν η πιθανότητα ύπαρξης ακμής είναι μεγάλη, ο κάθε κόμβος θα συνδέεται σχεδόν με όλους τους άλλου κόμβους

$p = 0.1$ $p = 0.25$ $p = 0.5$

2.2: Κεντρικότητες σύνθετων δικτύων

2.2.1: Degree centrality

Η κεντρικότητα βαθμού αποτελεί την πιο απλή κεντρικότητα των σύνθετων δικτύων. Η degree centrality εκφράζει το πλήθος των ακμών ενός κόμβου και κατά συνέπεια πόσους γείτονες έχει ο κάθε κόμβος. Στην περίπτωση που το δίκτυο είναι κατευθυνόμενο, η degree centrality του κάθε κόμβου χωρίζεται σε in-degree και out-degree. Η in-degree του κόμβου u υπολογίζει πόσες ακμές δείχνουν τον κόμβο u και η out-degree πόσους κόμβους δείχνει ο κόμβος u.



2.2.2: Betweenness centrality

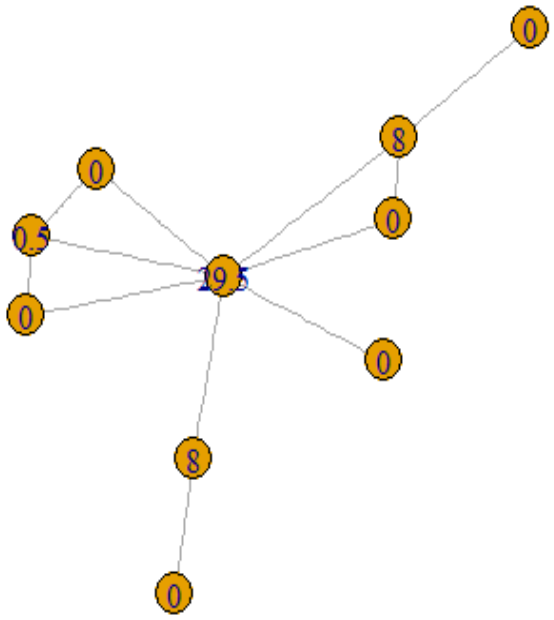
Η ενδιάμεση κεντρικότητα είναι μια ευρέως χρησιμοποιημένη κεντρικότητα, καθώς για τον υπολογισμό της χρησιμοποιούνται τα συντομότερα μονοπάτια, τα οποία κατέχουν κυρίαρχο ρόλο στα σύνθετα δίκτυα. Συγκεκριμένα, η betweenness centrality ενός κόμβου u υπολογίζεται ως εξής:

$$C_B(i) = \sum_{j<k} \frac{g_{jk}(i)}{g_{jk}},$$

Όπου,

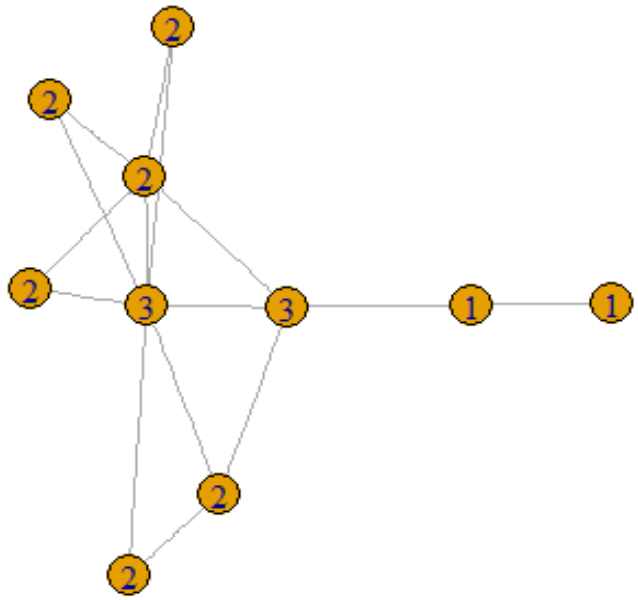
gjk είναι το πλήθος των συντομότερων μονοπατιών μεταξύ των κόμβων j και k, και

gjk(i) είναι το πλήθος των συντομότερων μονοπατιών μεταξύ των j και k που περνούν από τον κόμβο i



2.2.3: Power community index (PCI)

Η κεντρικότητα power community index αποτελεί μια ξεχωριστή κεντρικότητα, καθώς το PCI ενός κόμβου u δεν εξαρτάται από τον κόμβο u, αλλά από τους γείτονες του u. Συγκεκριμένα, το PCI ενός κόμβου u είναι ίσο με k, αν υπάρχουν k γείτονες του u, όπου ο καθένας τους έχει degree centrality μεγαλύτερο ή ίσο από k. Το PCI αναγνωρίζει του κόμβους που βρίσκονται στο πυκνό μέρος του δικτύου.

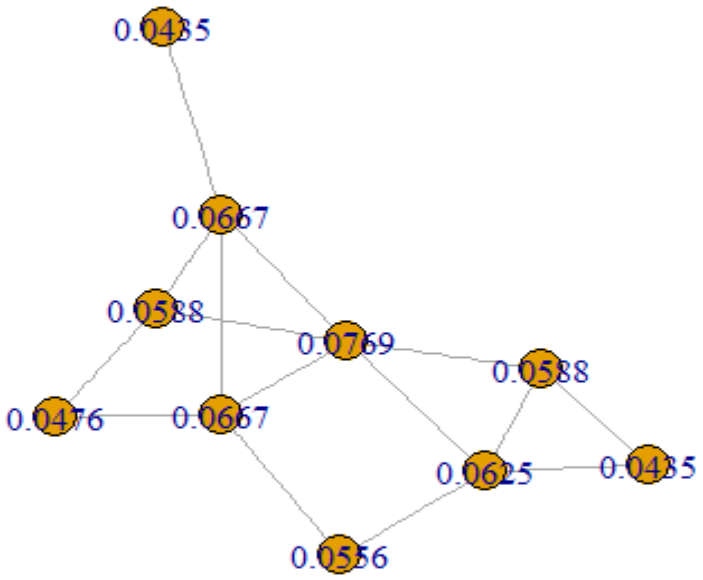


2.2.4: Closeness centrality

Η closeness centrality υπολογίζει πόσο κεντρικός είναι ένας κόμβος. Συγκεκριμένα, η closeness centrality ενός κόμβου u, υπολογίζεται ως το άθροισμα τους μήκους των συντομότερων μονοπατιών από τον κόμβο u προς τους υπόλοιπους κόμβους του δικτύου.

$$C_C(i) = \frac{n-1}{\sum_{j=1}^{n} d(i,j)}.$$

Όπου d(i, j) είναι η απόσταση του κόμβου I από τον κόμβο j.

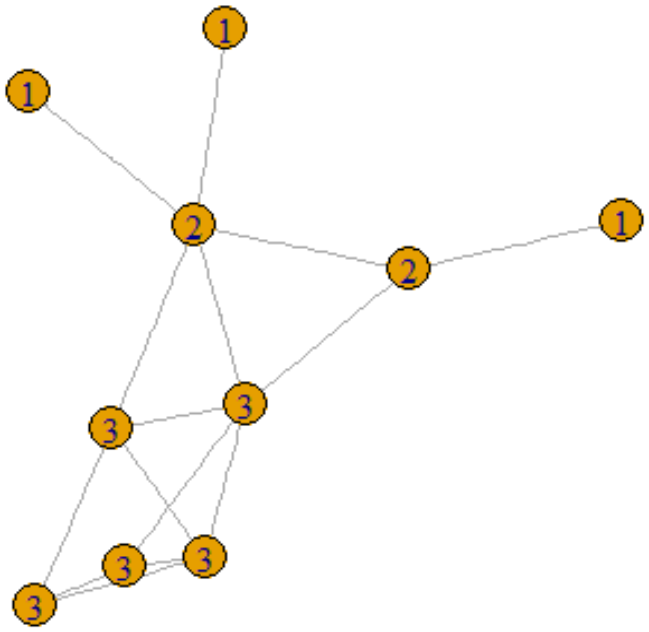


2.2.5: K Shell

Η k shell υπολογίζει σε πόσο πυκνό μέρος του δικτύου ανήκει ο κόμβος. Συγκεκριμένα το υποδίκτυο με kshell ίσο με k, είναι όλοι οι κόμβοι που με βαθμό ίσο με k. Ο αλγόριθμος υπολογισμού του k shell είναι ο εξής:

1. Αφαίρεσε από το δίκτυο όλους τους κόμβους με degree ίσο με 1.
2. Κάποιοι εναπομείναντες κόμβοι ίσως έχουν degree ίσο με 1. Επανάλαβε μέχρι να μην υπάρχει κανένας κόμβος με degree ίσο με 1.
3. Το υπόλοιπο δίκτυο αποτελεί το 2-shell δίκτυο.
4. Επανάλαβε μέχρι να βρεις όλα τα k-shell.

Οποιοδήποτε δίκτυο μπορεί να εκφραστεί ως ένα σύνολο από k-shells



Κεφάλαιο 3

Αλγόριθμος επίλυσης

Σε αυτό το κεφάλαιο παρουσιάζεται ο αλγόριθμος επίλυσης του προβλήματος των ελάχιστων κόμβων καταχώρησης. Παρουσιάζεται ο ψευδοκώδικας και στη συνέχεια η υλοποίηση της εργασίας στη γλώσσα R χρησιμοποιώντας τη βιβλιοθήκη igraph.

2.1: Ψευδοκώδικας

Για να λύσουμε το πρόβλημα των ελάχιστων κόμβων καταχώρησης θα χρησιμοποιήσουμε ως βασικό εργαλείο τα συντομότερα μονοπάτια, καθώς αποτελούν κυρίαρχο ρόλο στην επιστήμη των σύνθετων δικτύων. Όπως αναφέρθηκε και στα Κεφάλαιο 1, έχουμε μια και μοναδική προϋπόθεση. Κάθε συντομότερο μονοπάτι θα περιέχει τουλάχιστον ένα κόμβο καταχώρησης. Με αυτό τον τρόπο θα εξασφαλίσουμε ότι οι κόμβοι καταχώρησης θα μπορούν να καλύψουν ολόκληρο το δίκτυο. Με βάση τα παραπάνω ο αλγόριθμος επίλυσης του προβλήματος ελάχιστων κόμβων καταχώρησης, αποτελείται από τα εξής βήματα/στάδια.

1. Βρες τα συντομότερα μονοπάτια του δικτύου.
2. Ταξινόμησε τους κόμβους του δικτύου κατά φθίνουσα σειρά με βάση κάποια κεντρικότητα των σύνθετων δικτύων. (degree centrality, betweenness centrality, closeness centrality, PCI, kshell)
3. Για κάθε ένα ταξινομημένο κόμβο, βρες πόσα συντομότερα μονοπάτια περνάνε από αυτόν τον κόμβο και πρόσθεσέ τα σε ένα σύνολο. Κατά την πρόσθεση, πρόσεξε για διπλότυπα.
4. Για κάθε έναν ταξινομημένο κόμβο βρες το συνολικό ποσοστό του δικτύου που έχει καλυφθεί δηλαδή πόσα shortest paths έχεις καλύψει.
5. Βρες πόσους κόμβους χρειάστηκε η κάθε κεντρικότητα για να καλύψει όλο το δίκτυο.

Μας ενδιαφέρει να βρούμε την κεντρικότητα με την οποία θα έχουμε τους ελάχιστους κόμβους καταχώρησης. Αυτό συνεπάγεται πως για κάθε ταξινομημένο κόμβο του δικτύου, το συνολικό ποσοστό θα αυξάνεται (αρκετά) και είναι πολύ πιθανό να μην χρειαστούμε όλους τους κόμβους του δικτύου για να καλύψουμε όλο το δίκτυο. Θα εκτυπώσουμε τα αποτελέσματα του παραπάνω αλγόριθμου και θα κάνουμε τις γραφικές παραστάσεις (cover rate function) για κάθε κεντρικότητα. Περιμένουμε η κλίση της γραφικής παράστασης να είναι (αρκετά) απότομη στην αρχή.

Ακολουθεί ο ψευδοκώδικας επίλυσης του προβλήματος των ελάχιστων κόμβων καταχώρησης

```
Centralities = [ degree, betweenness, closeness, PCI, k-Shell]
allShortestPaths = findTheShortestPaths( complexNetwork)
for( c in centralities) {
        sortedNodes = sort( c)
        for( n in sortedNodes){
                percentage = calculatePartialShortestPaths( n)
        }
        coverRateFunction = percentage
        calculateCheckInNodes( c)
}
```

2.2: Υλοποίηση

Για την υλοποίηση του αλγόριθμου χρησιμοποιήθηκε η γλώσσα R και η βιβλιοθήκη igraph.

Η πρώτη υλοποίηση ήταν η εξής:

1. Βρες όλα τα shortest paths και αποθήκευσε τα σε μια λίστα (allShortestPaths).
2. Για κάθε ένα κόμβο βρες τα shortest paths του κόμβου προς όλους τους άλλους κόμβους του δικτύου. Αποθήκευσε τα σε μια λίστα (NodeShortestPaths).
3. Βρες όλα τα shortest paths από τους άλλους κόμβους προς αυτόν τον κόμβο. Πρόσθεσε τα στην ιδιά λίστα(NodeShortestPaths).
4. Τρέξε την λίστα με όλα τα shortest paths του δικτύου. Αν υπάρχει shortest path που περνάει από αυτόν τον κόμβο πρόσθεσέ το στην λίστα(NodeShortestPaths), αφού πρώτα ελέγξεις ότι δεν είναι ήδη αποθηκευμένο. Πλέον έχεις όλα τα shortest paths που περνούν από αυτόν τον κόμβο.
5. Στη συνέχεια πρόσθεσε κάθε ένα στοιχείο αυτής της λίστας σε μια άλλη λίστα (partialShortestPaths) όπου θυμάσαι ποσά συνολικά shortests paths έχεις δει, αφού πρώτα κάνεις έλεγχο αν το shortest path υπάρχει ήδη στη λίστα.
6. Διαίρεσε το πλήθος των δυο λιστών, της λίστας που έχει όλα τα shortest paths (allShortestPaths) του δικτύου και την λίστα με τα shortest paths που έχουμε δει μέχρι στιγμής (partialShortestPaths), για να βρεις τι ποσοστό του δικτύου έχουμε καλύψει. Και έτσι υπολογίζουμε την cover rate function.
7. Βρες για κάθε μια κεντρικότητα πόσοι κόμβοι χρειάστηκαν για να καλύψεις όλο το δίκτυο.
8. Επανάλαβε την ίδια διαδικασία και για τις υπόλοιπες κεντρικότητες.

Αυτή η υλοποίηση ήταν σωστή και επέστρεφε σωστά αποτελέσματα, αλλά ιδιαίτερα αργή, ειδικά για μεγάλα δίκτυα. Πραγματοποιούνταν (αρκετές) αναθέσεις τιμών και (αρκετοί) έλεγχοι χωρίς να χρειάζεται. Για παράδειγμα, όταν έμεναν οι τελευταίοι ταξινομημένοι κατά μια κεντρικότητα κόμβοι για έλεγχο, ο αλγόριθμος διέτρεχε χωρίς λόγο τη λίστα με όλα τα shortests paths του δικτύου. Τα περισσότερα από αυτά τα shortest paths τα έχω προσθέσει ήδη στη λίστα με τα shortest paths που έχω δει μέχρι στιγμής (partialShortestPaths).

Ψευδοκώδικας πρώτης υλοποίησης

```
Centralities = ( betweenness, degree, closeness, PCI, kshell)
allShortestPaths = findAllShortestPaths( network)
for( c in centralities) {
        sortedNodes = sort( c)
        for( n in sortedNodes) {
                nodeShortestPaths = add( inShortestPaths( n))
                nodeShortestPaths = add( outShortestPaths( n))
                for( sPath in allShortestPaths) {
                        if( usesNode( sPath, n)) {
                                if( !exists( sPath, nodeShortestPaths)) {
                                        nodeShortestPaths = add( sPath)
                                }
                        }
                }
                For( sPath in nodeShortestPaths) {
                        If( !exists( sPath, partialShortestPaths) {
                                partialShortestPaths = add( sPath)
                        }
                }
                Percentage = length(partialShortestPaths) / length(allShortestPaths)
        }
        numberOfCheckInNodes = findNumberOfCheckInNodes( percentage)
}
```

Η δεύτερη υλοποίηση ήταν η εξής:

1. Βρες όλα τα shortest paths του δικτύου και αποθήκευσε τα σε μια λίστα (allShortestPaths).
2. Δημιούργησε ένα αντίγραφο της (allShortestPathsCopy).
3. Για κάθε ένα κόμβο του δικτύου διέτρεξε το αντίγραφο που περιέχει όλα τα shortest paths του δικτύου. Αν βρεις κάποιο shortest path που περνά από τον κόμβο, αποθήκευσε σε μια λίστα τη θέση του (nodeShortestPaths).
4. Για κάθε ένα στοιχείο αυτής της λίστας αύξησε ένα μετρητή κατά μια μονάδα και αφαίρεσε από το αντίγραφο τα shortest paths των συγκεκριμένων θέσεων.
5. Διαίρεσε τον μετρητή με το μήκος της λίστας που έχει όλα τα shortest paths (allShortestPaths) για να βρεις το ποσοστό του δικτύου που έχεις καλύψει, υπολόγισε δηλαδή την cover rate function.
6. Υπολόγισε πόσες κόμβους χρειάστηκε η κάθε κεντρικότητα για να καλύψει όλο το δίκτυο, δηλαδή το πλήθος των check in nodes.
7. Επανάλαβε και για τις υπόλοιπες κεντρικότητες.

Η δεύτερη υλοποίηση ήταν αισθητά πιο γρήγορη καθώς οι αναθέσεις τιμών και οι έλεγχοι μειωθήκαν δραματικά. Πλέον για κάθε επόμενο ταξινομημένο κατά μια κεντρικότητα κόμβο η λίστα που θα ελέγχεται θα είναι μικρότερη από την προηγουμένη φορά και ξέρω ότι οποίο shortest path βρω ότι περνάει από τον συγκεκριμένο κόμβο μπορεί να βοηθήσει στην αύξηση της cover rate function. Δεν χρειάζεται πλέον να κάνω έλεγχο για διπλότυπα. Μια ακόμα βελτίωση που έγινε σε αυτή την υλοποίηση ήταν η δέσμευση μνήμης για τις λίστες πριν την εκτέλεση, καθώς στην γλώσσα R η συνεχής αύξηση μεγέθους μας λίστας είναι αρκετά χρονοβόρα.

Ψευδοκώδικας δεύτερης υλοποίησης

```
Centralities = ( betweenness, degree, closeness, PCI, kshell)

allShortestPaths = findAllShortestPaths( network)

For( c in centralities) {

        sortedNodes = sort( c)

        allShortestPathsCopy = allShortestPaths

        counter = 0

        for( n in sortedNodes) {

                for( sPath in allShortestPathsCopy) {

                        if( usesNode( sPath, n) {

                                nodeShortestPaths = add( index( allShortestPaths[ sPath] )

                        }

                }

                For( i in nodeShortestPaths) {

                        counter = counter + 1

                        remove( allShortestPathsCopy, nodeShortestPaths[ i])

                }

                Percentage = counter / length( allShortestPaths)

        }

        numberOfCheckInNodes = findNumberOfCheckInNodes( percentage)

}
```

Κεφάλαιο 4

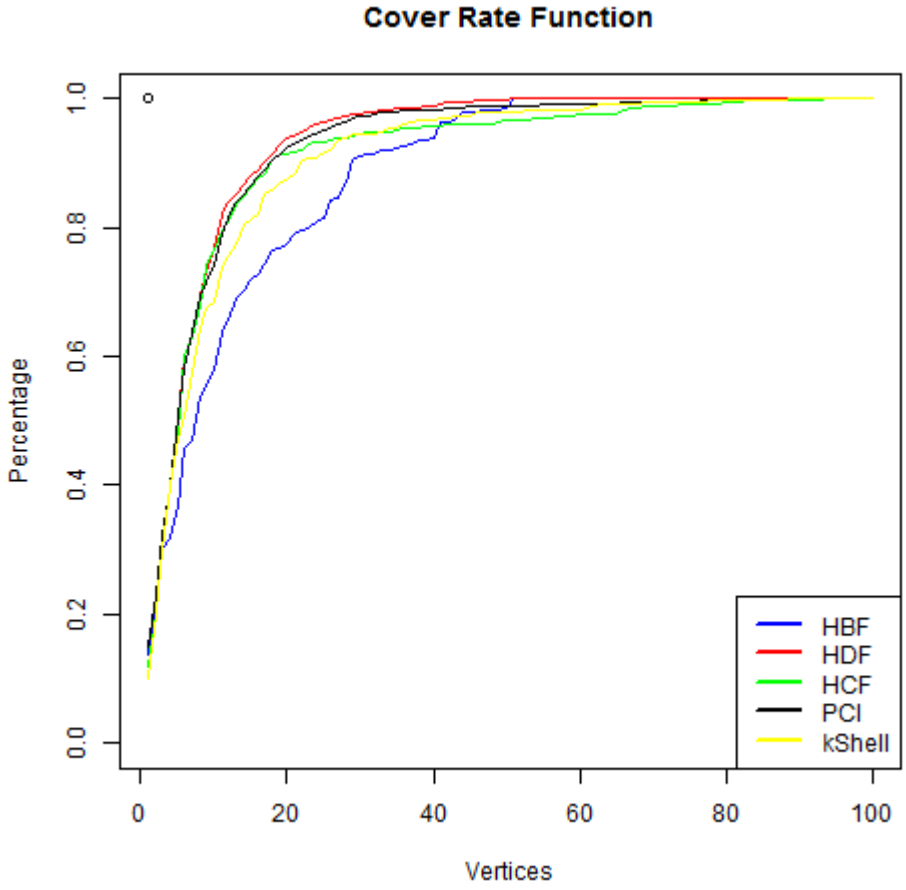
Προσομοιώσεις

Τα παρακάτω αποτελέσματα είναι οι μέσες τιμές πέντε προσομοιώσεων. Προφανώς, όσο περισσότερες οι προσομοιώσεις, τόσο πιο «καθαροί» αριθμοί θα προκύψουν.

4.1: Scale free δίκτυα (Barabási–Albert μοντέλο)

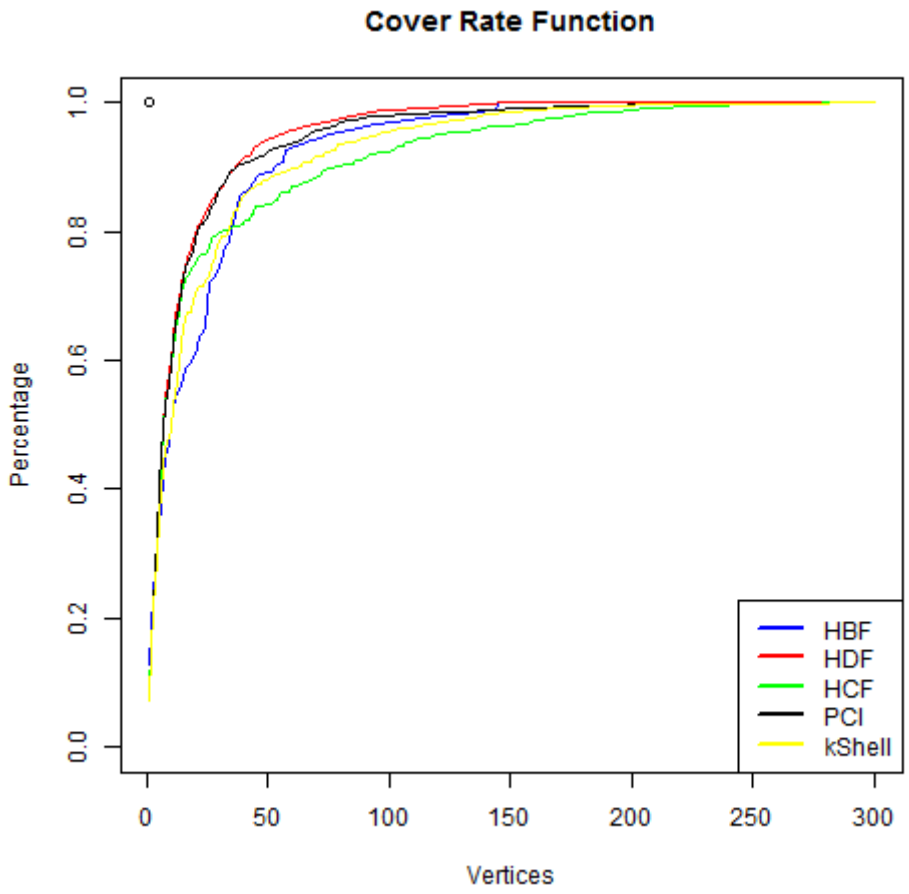
4.1.1: Κατευθυνόμενα δίκτυα

N = 100



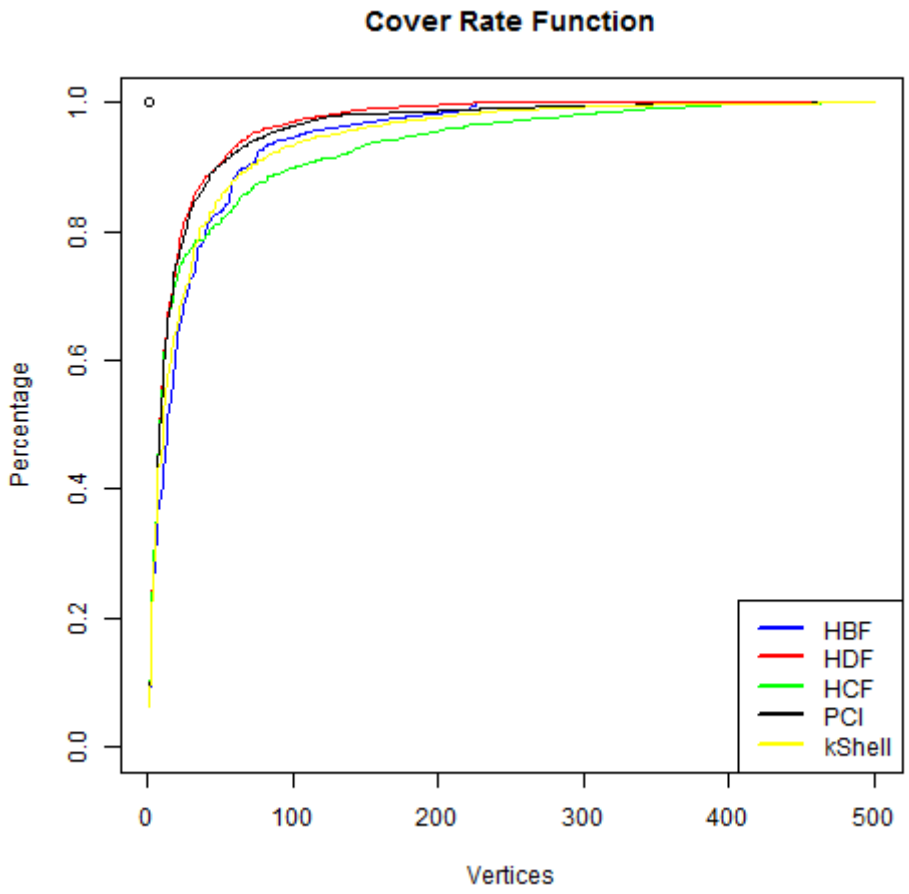
Centrality	Number of check-in nodes	Percentage
Betweenness	51	51/100 = 51 %
Degree	51	51/100 = 51 %
Closeness	94	94/100 = 94 %
PCI	89	89/100 = 89 %
kShell	89	89/100 = 89 %

N = 300



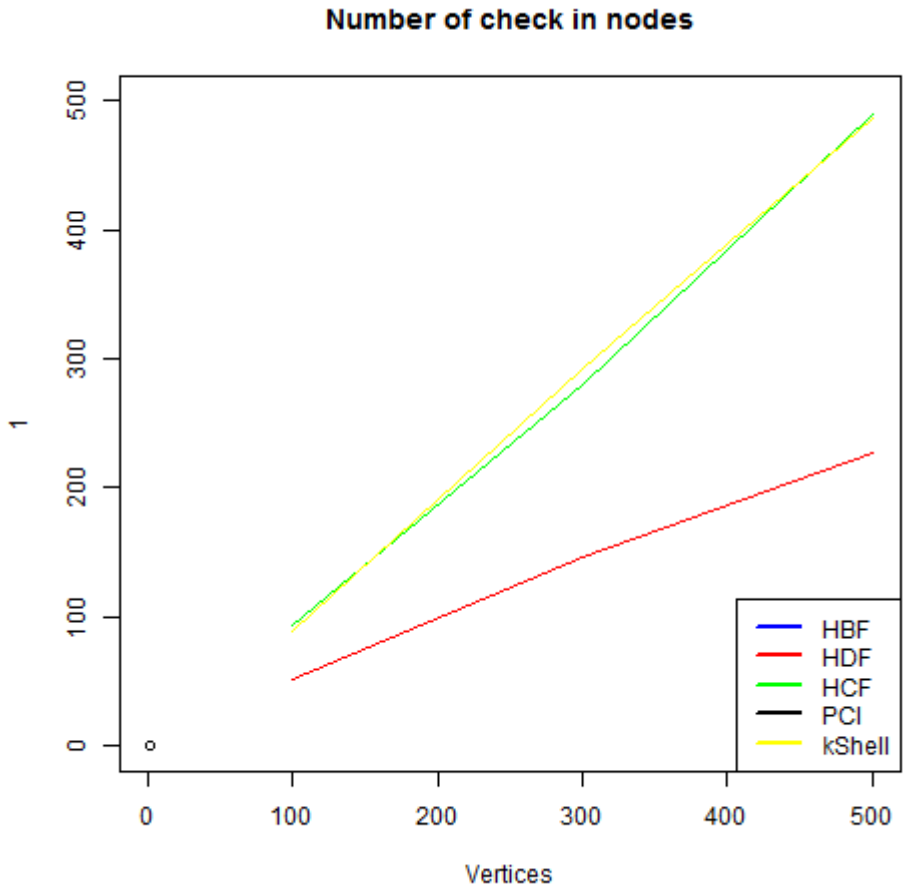
Centrality	Number of check-in nodes	Percentage
Betweenness	146	146 / 300 = 48,6 %
Degree	146	146 / 300 = 48,6 %
Closeness	280	280 / 300 = 93,3 %
PCI	293	293 / 300 = 97,6 %
kShell	293	293 / 300 = 97,6 %

N = 500



Centrality	Number of check-in nodes	Percentage
Betweenness	227	227 / 500 = 45,4 %
Degree	227	227 / 500 = 45,4 %
Closeness	489	489 / 500 = 97,8 %
PCI	487	487 / 500 = 97,4 %
kShell	487	487 / 500 = 97,4 %

Διάγραμμα πλήθους check-in κόμβων για κατευθυνόμενα scale free δίκτυα 100, 300, 500 κόμβων

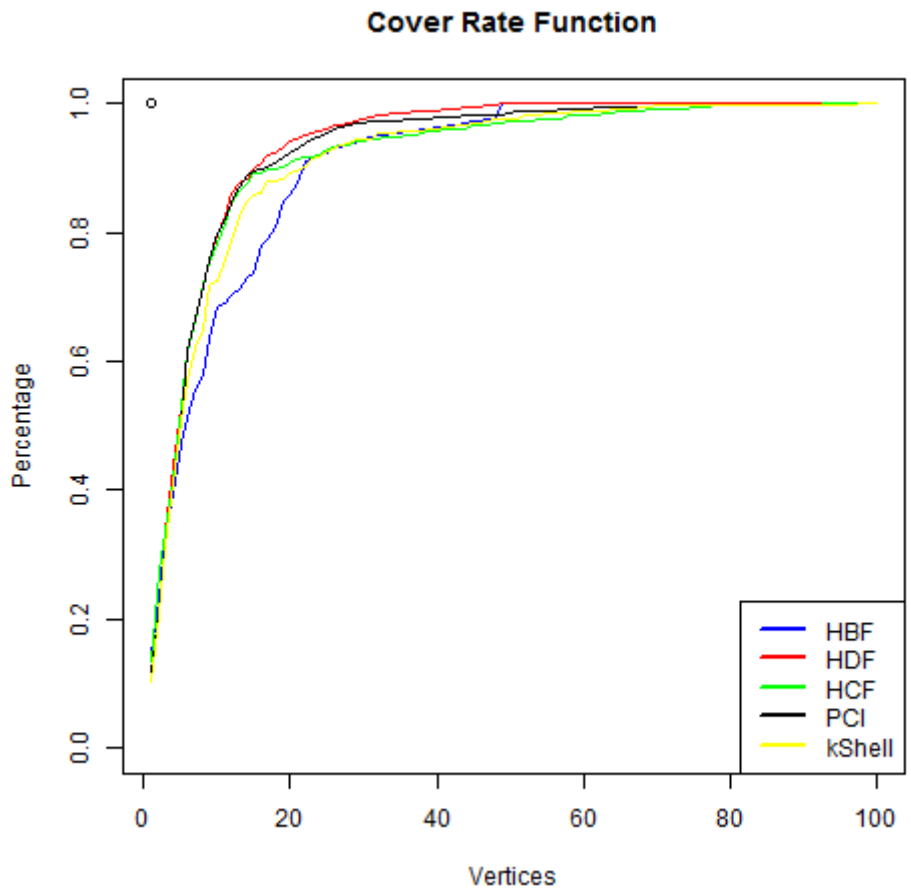


Οι κεντρικότητες εκτυπώνονται με την εξής σειρά: Betweenness centrality, degree centrality, closeness centrality, PCI, Kshell. Επειδή και στα τρία μεγέθη δικτύων οι betweenness και η degree centrality έχουν ως αποτέλεσμα το ίδιο πλήθος check-in κόμβων, η degree centrality επικαλύπτει την betweenness στο διάγραμμα. Το ίδιο συμβαίνει και για το PCI με το kShell.

Συγκρίνοντας τις κεντρικότητες παρατηρούμε πως οι betweenness και οι degree centrality έχουν καλύτερα αποτελέσματα καθώς χρειάζονται περίπου τους μισούς κόμβους του δικτύου για να καλύψουν όλα τα συντομότερα μονοπάτια του δικτύου και επομένως ολόκληρο το δίκτυο. Η closeness, το PCI και το kShell είναι σχεδόν ίδια μεταξύ τους και χρειάζονται κάτι λιγότερο από ολόκληρο το δίκτυο.

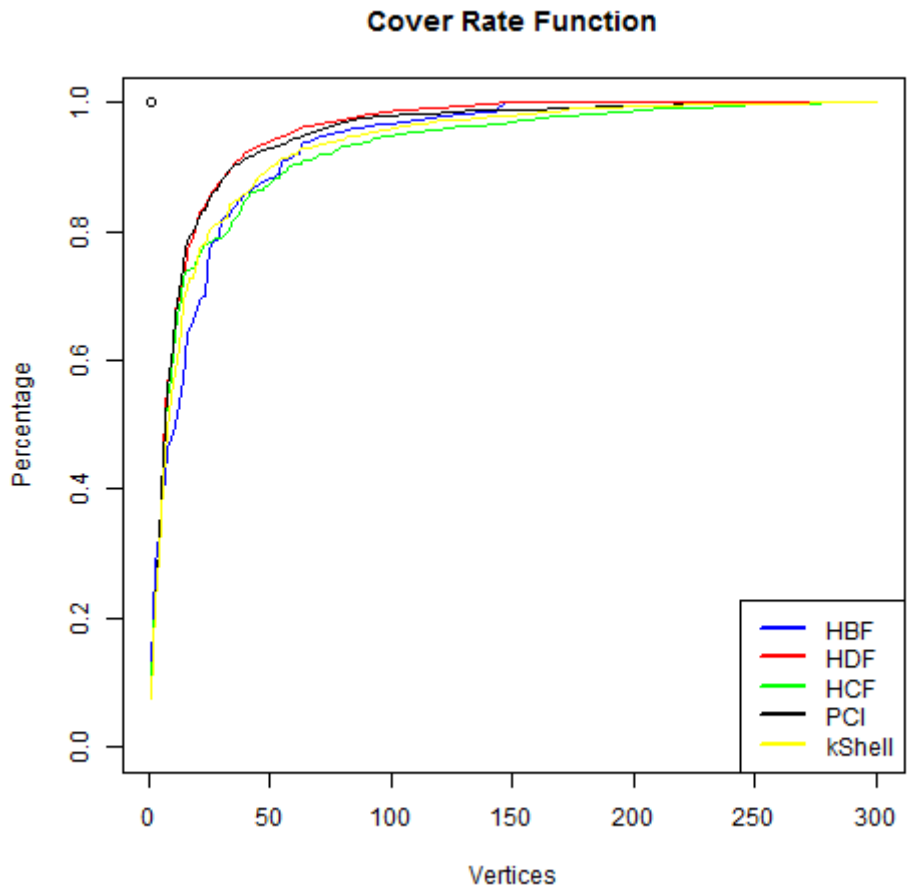
4.1.2: Μη κατευθυνόμενα δίκτυα

N = 100



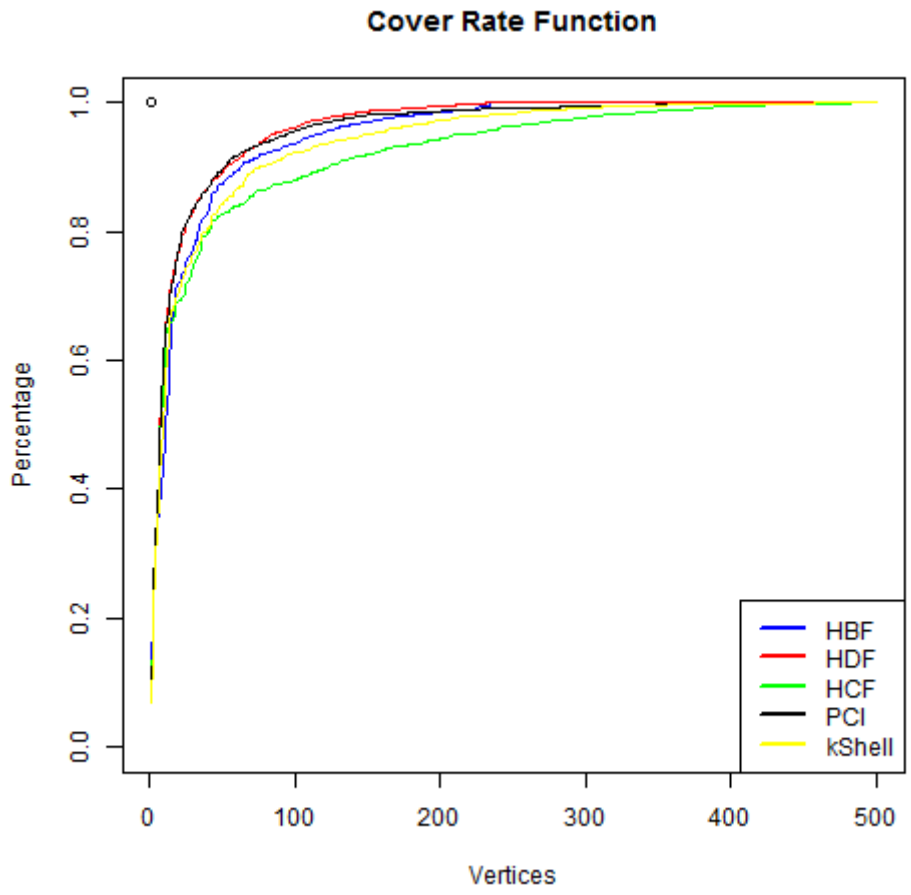
Centrality	Number of check-in nodes	Percentage
Betweenness	49	49 / 100 = 49 %
Degree	49	49 / 100 = 49 %
Closeness	93	93 / 100 = 93 %
PCI	98	98 / 100 = 98 %
kShell	98	98 / 100 = 98 %

N = 300



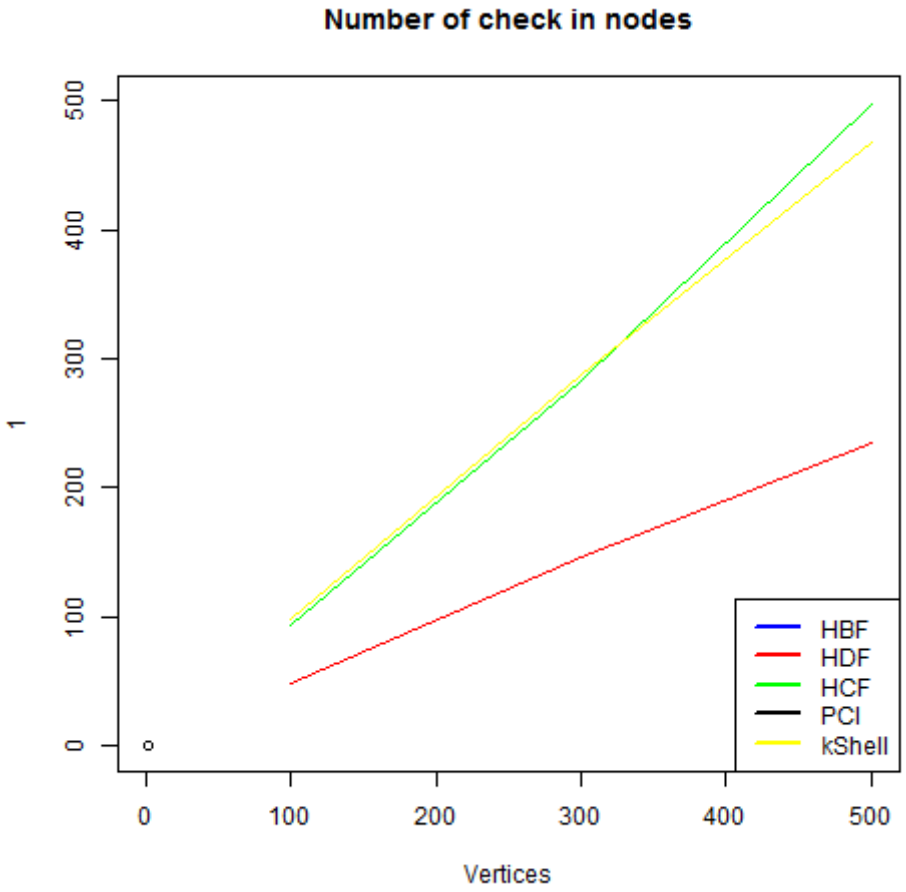
Centrality	Number of check-in nodes	Percentage
Betweenness	147	147 / 300 = 49 %
Degree	147	147 / 300 = 49 %
Closeness	283	283 / 300 = 94,3 %
PCI	287	287 / 300 = 95,6 %
kShell	287	287 / 300 = 95,6 %

N = 500



Centrality	Number of check-in nodes	Percentage
Betweenness	235	235 / 500 = 47 %
Degree	235	235 / 500 = 47 %
Closeness	498	498 / 500 = 99,6 %
PCI	468	468 / 500 = 93,6 %
kShell	468	468 / 500 = 93,6 %

Διάγραμμα πλήθους check-in κόμβων για μη κατευθυνόμενα scale free δίκτυα 100, 300, 500 κόμβων



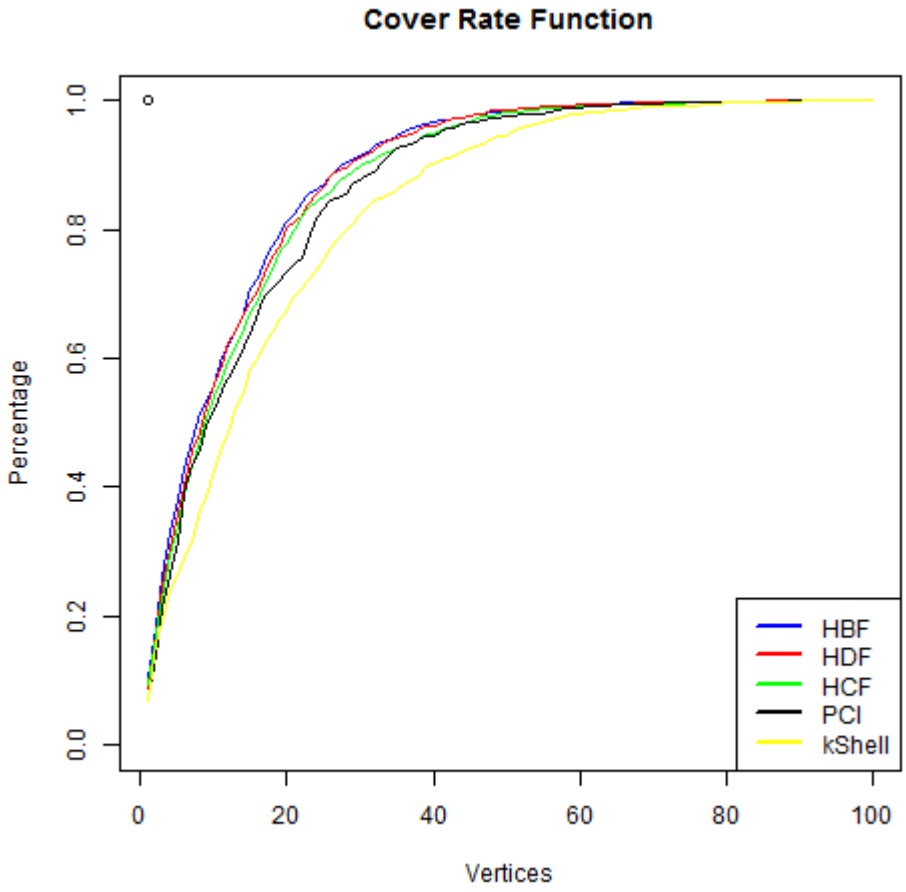
Όπως και στα κατευθυνόμενα δίκτυα, οι κεντρικότητες εκτυπώνονται με την εξής σειρά: Betweenness centrality, degree centrality, closeness centrality, PCI, Kshell. Επειδή και στα τρία μεγέθη δικτύων οι betweenness και η degree centrality έχουν ως αποτέλεσμα το ίδιο πλήθος check-in κόμβων, η degree centrality επικαλύπτει την betweenness στο διάγραμμα. Το ίδιο συμβαίνει και για το PCI με το kShell.

Όπως και στα κατευθυνόμενα δίκτυα παρατηρούμε ότι οι betweenness και η degree centrality χρειάζονται περίπου τους κόμβους του δικτύου για να καλύψουν όλα τα συντομότερα μονοπάτια και επομένως ολόκληρο το δίκτυο. Η closeness, το PCI και το kShell επίσης είναι σχεδόν ίδια μεταξύ τους και πάλι χρειάζονται κάτι λιγότερο από όλο το δίκτυο.

4.2: Random δίκτυα (Μοντέλο Erdős–Rényi)

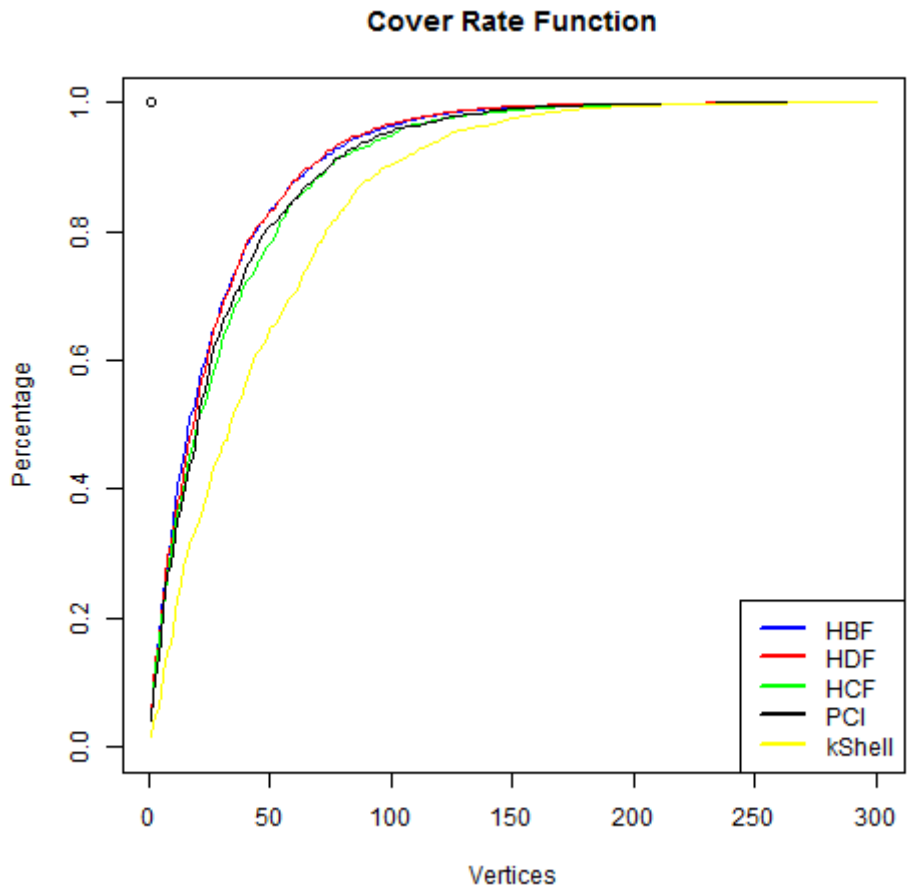
4.2.1: Κατευθυνόμενα δίκτυα

N = 100



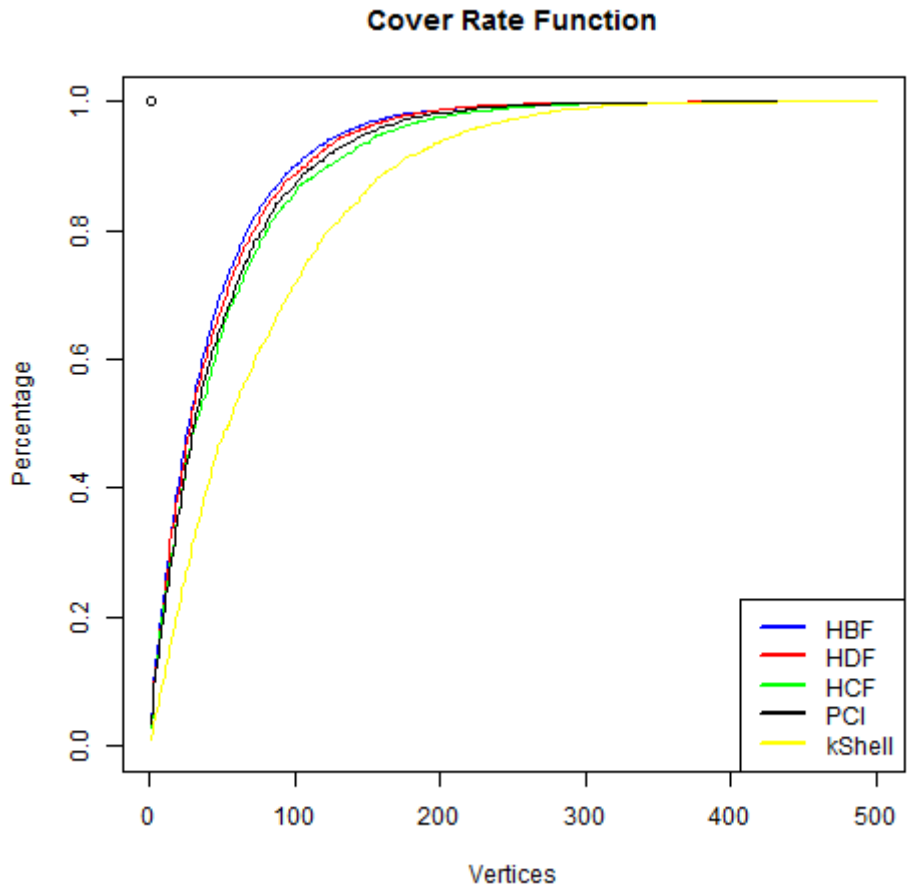
Centrality	Number of check-in nodes	Percentage
Betweenness	96	96 / 100 = 96 %
Degree	94	94 / 100 = 94 %
Closeness	94	94 / 100 = 94 %
PCI	96	96 / 100 = 96 %
kShell	95	95 / 100 = 95 %

N = 300



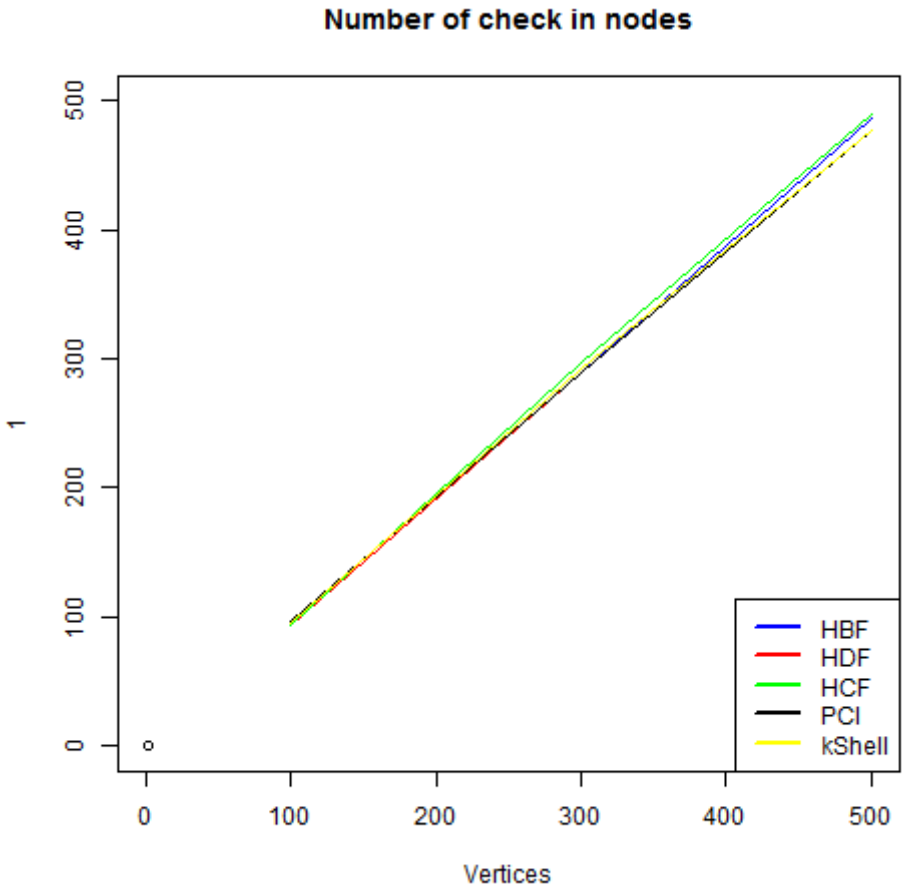
Centrality	Number of check-in nodes	Percentage
Betweenness	289	289 / 300 = 96,3 %
Degree	290	290 / 300 = 96,6 %
Closeness	297	297 / 300 = 99 %
PCI	290	290 / 300 = 96,6 %
kShell	292	292 / 300 = 97,3 %

N = 500



Centrality	Number of check-in nodes	Percentage
Betweenness	486	486 / 500 = 97,2 %
Degree	477	477 / 500 = 95,4 %
Closeness	489	489 / 500 = 97,8 %
PCI	477	477 / 500 = 95,4 %
kShell	478	478 / 500 = 95,6 %

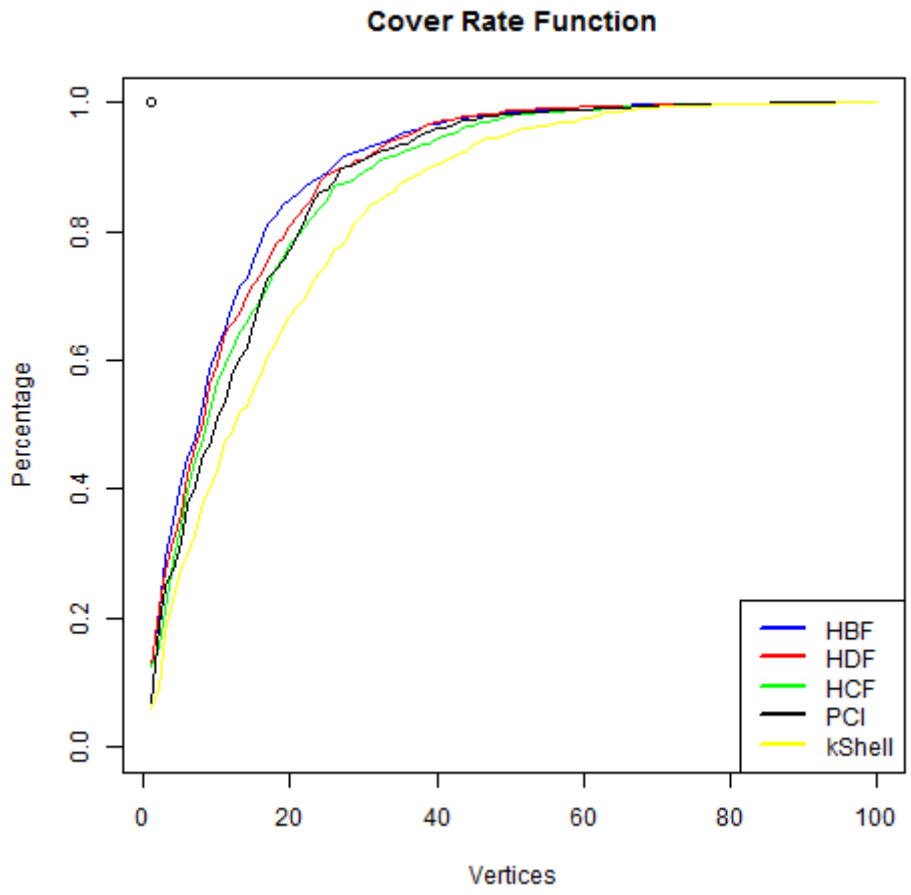
Διάγραμμα πλήθους check-in κόμβων για κατευθυνόμενα random δίκτυα 100, 300, 500 κόμβων



Παρατηρούμε ότι όλες οι κεντρικότητες είναι σχεδόν ίδιες μεταξύ τους και χρειάζονται σχεδόν όλους τους κόμβους του δικτύου για να καλύψουν όλα τα συντομότερα μονοπάτια.

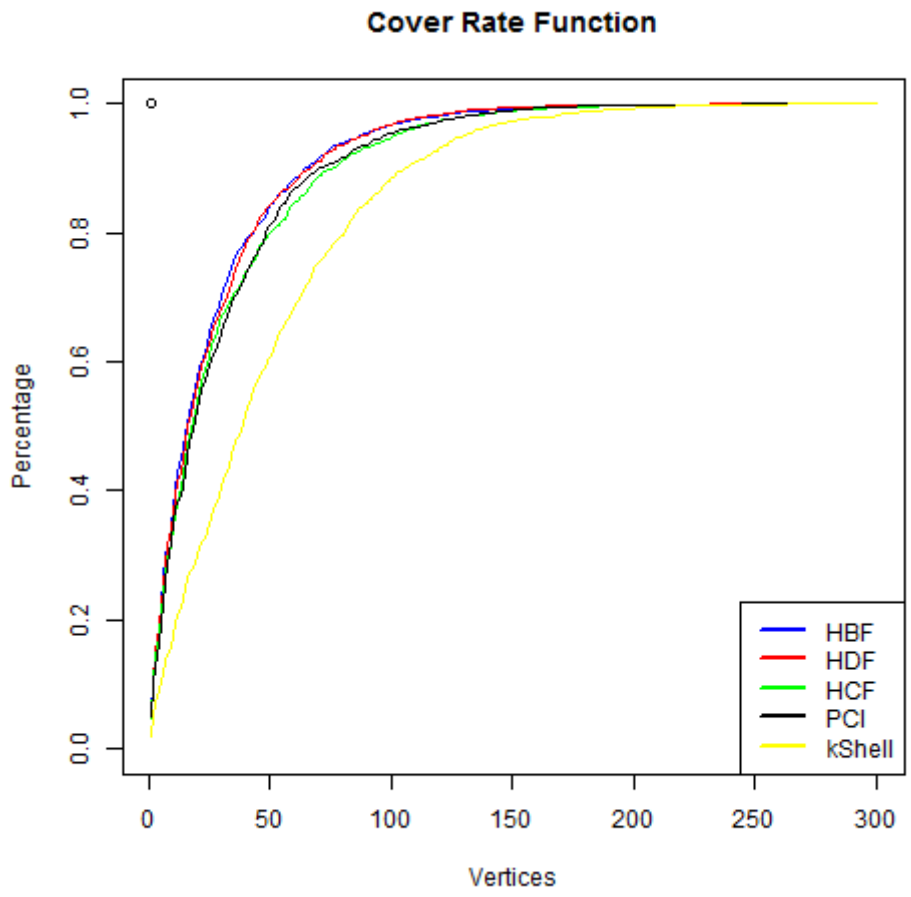
4.2.2: Μη κατευθυνόμενα δίκτυα

N = 100



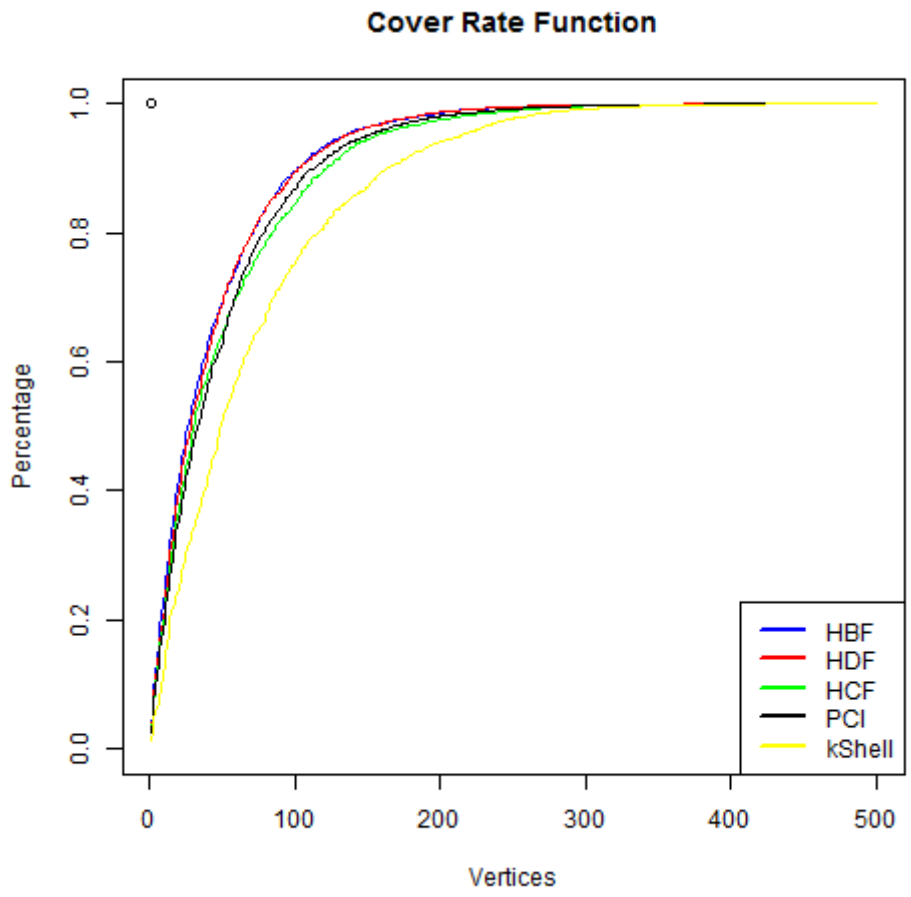
Centrality	Number of check-in nodes	Percentage
Betweenness	95	95 / 100 = 95 %
Degree	89	89 / 100 = 89 %
Closeness	97	97 / 100 = 97 %
PCI	97	97 / 100 = 97 %
kShell	98	98 / 100 = 98 %

N = 300



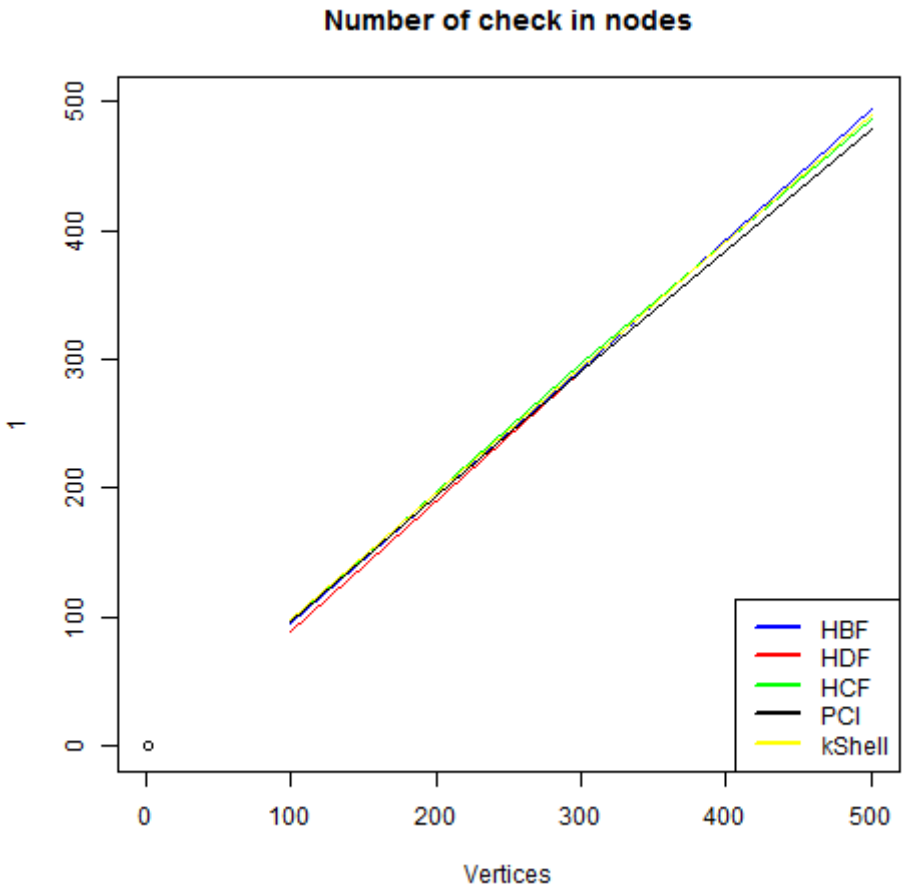
Centrality	Number of check-in nodes	Percentage
Betweenness	292	292 / 300 = 97,3 %
Degree	291	291 / 300 = 97 %
Closeness	297	297 / 300 = 99 %
PCI	291	291 / 300 = 97 %
kShell	294	294 / 300 = 98 %

N = 500



Centrality	Number of check-in nodes	Percentage
Betweenness	495	495 / 500 = 99 %
Degree	479	479 / 500 = 95, 8 %
Closeness	486	486 / 500 = 97,2 %
PCI	479	479 / 500 = 95,8 %
kShell	490	490 / 500 = 98 %

Διάγραμμα πλήθους check-in κόμβων για μη κατευθυνόμενα random δίκτυα 100, 300, 500 κόμβων



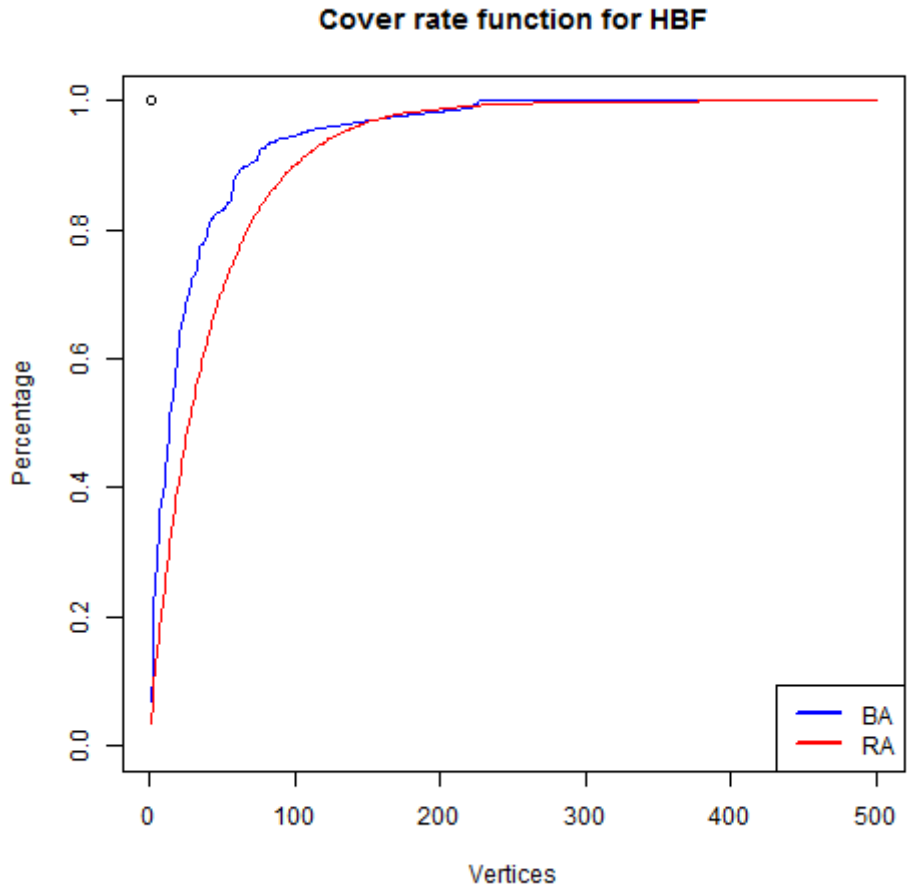
Όπως και στα κατευθυνόμενα δίκτυα παρατηρούμε πως όλες οι κεντρικότητες μοιάζουν μεταξύ τους και χρειάζονται σχεδόν όλους τους κόμβους του δικτύου για να καλύψουν όλα τα συντομότερα μονοπάτια.

4.3: Σύγκριση κεντρικοτήτων για τους τύπους δικτύων

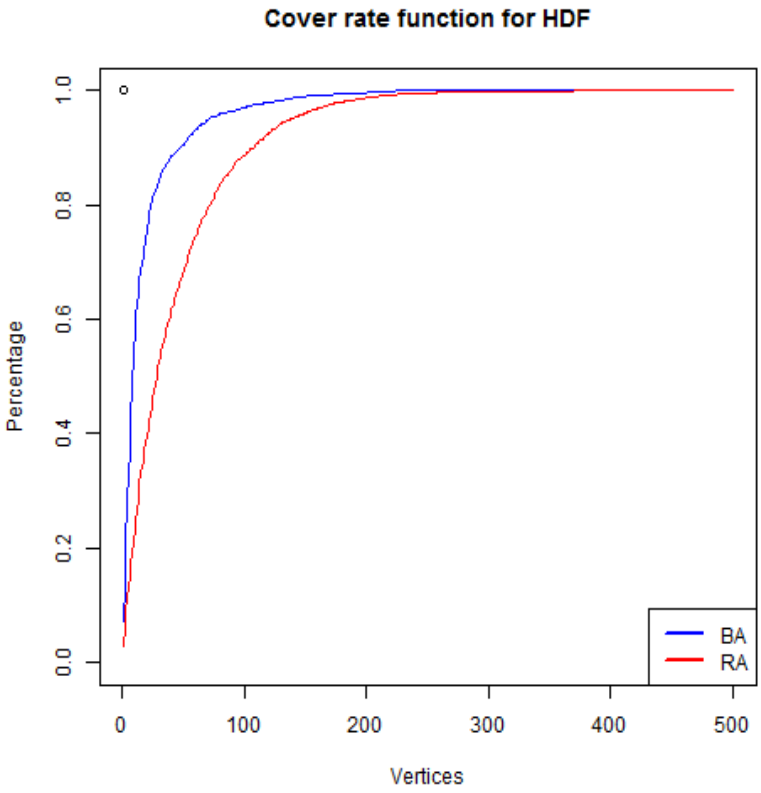
Σε αυτή τη παράγραφο θα συγκρίνουμε τις cover rate functions για κάθε μια κεντρικότητα για scale free και random δίκτυα, για κατευθυνόμενα και μη κατευθυνόμενα και για όλα τα μεγέθη δικτύων. Εδώ θα παρουσιαστούν για λόγους απλότητας τα δίκτυα 500 κόμβων. Τα αποτελέσματα και για τα τρία μεγέθη δικτύων βρίσκονται στο παράρτημα Α.

4.3.1: Κατευθυνόμενα δίκτυα

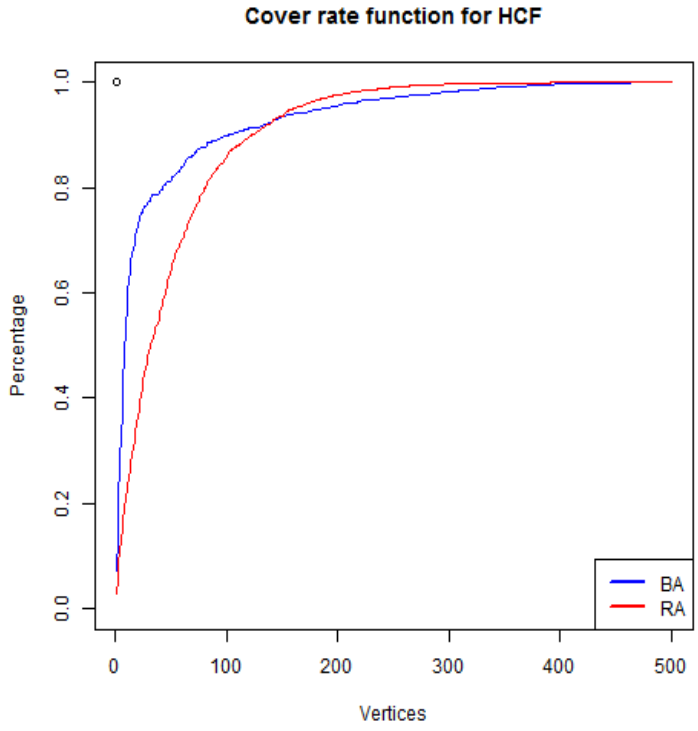
Betweenness centrality



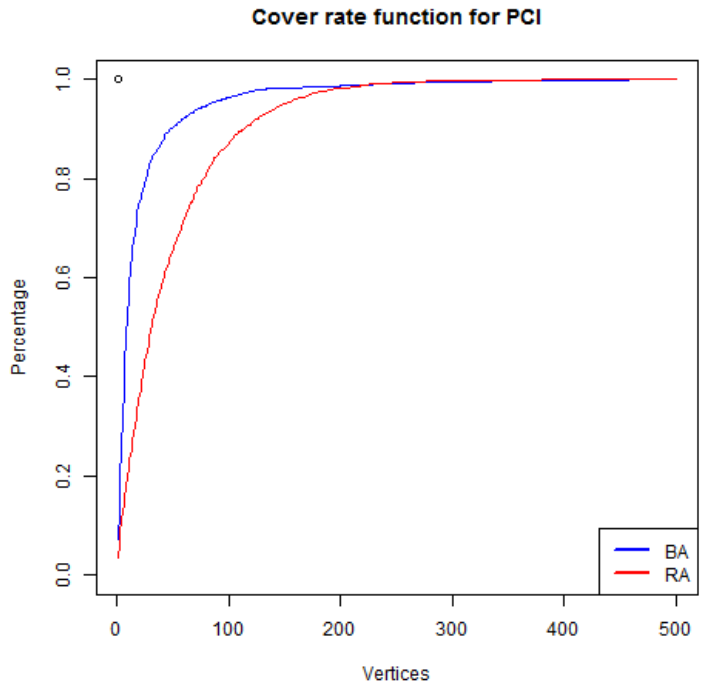
Degree Centrality



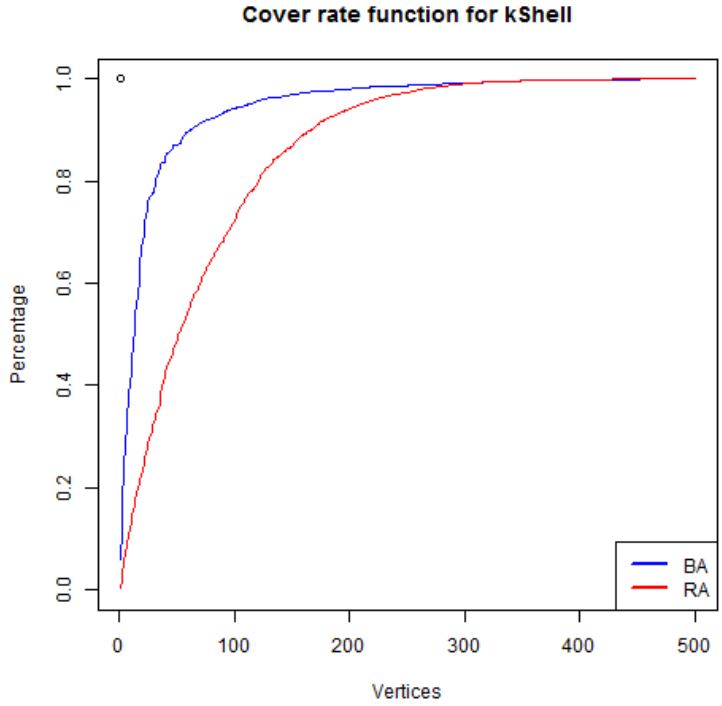
Closeness Centrality



PCI

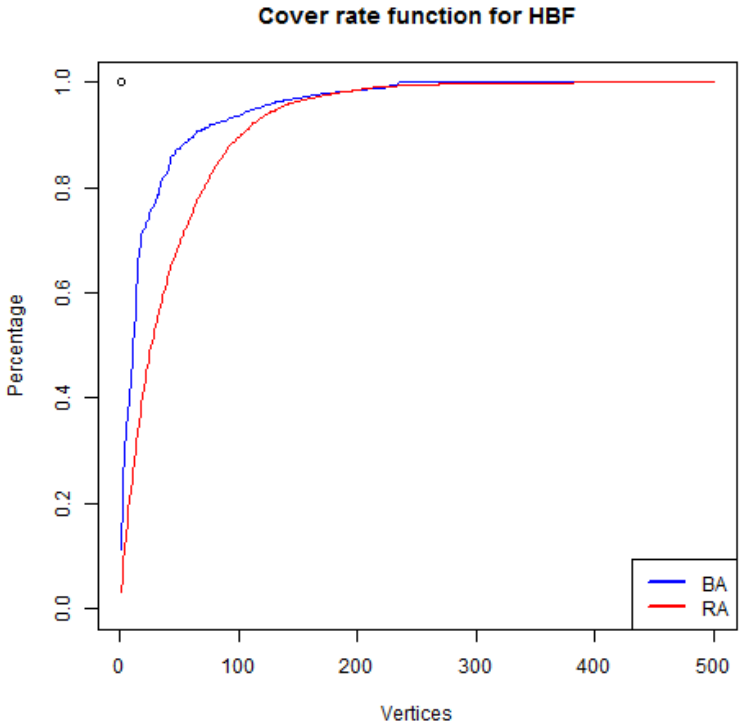


kShell

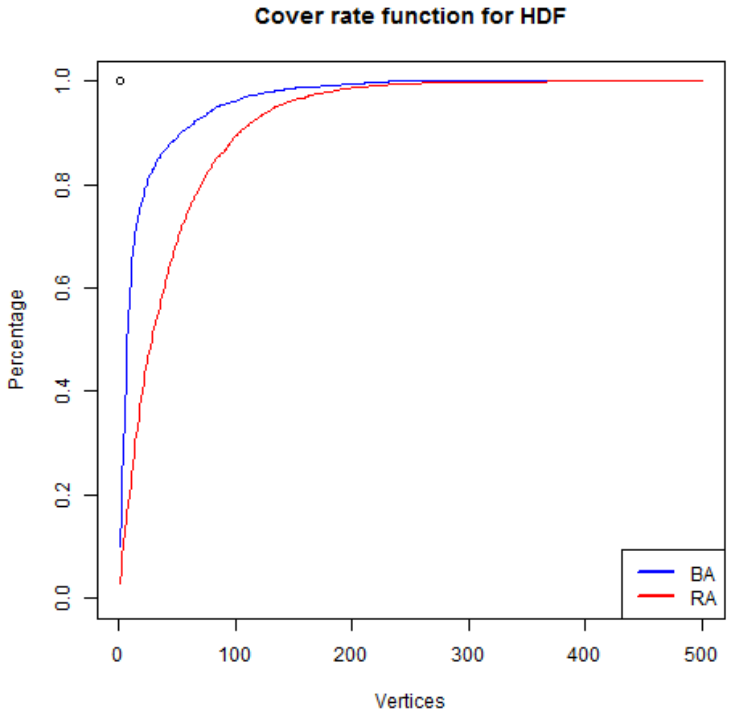


4.3. Μη κατευθυνόμενα δίκτυα

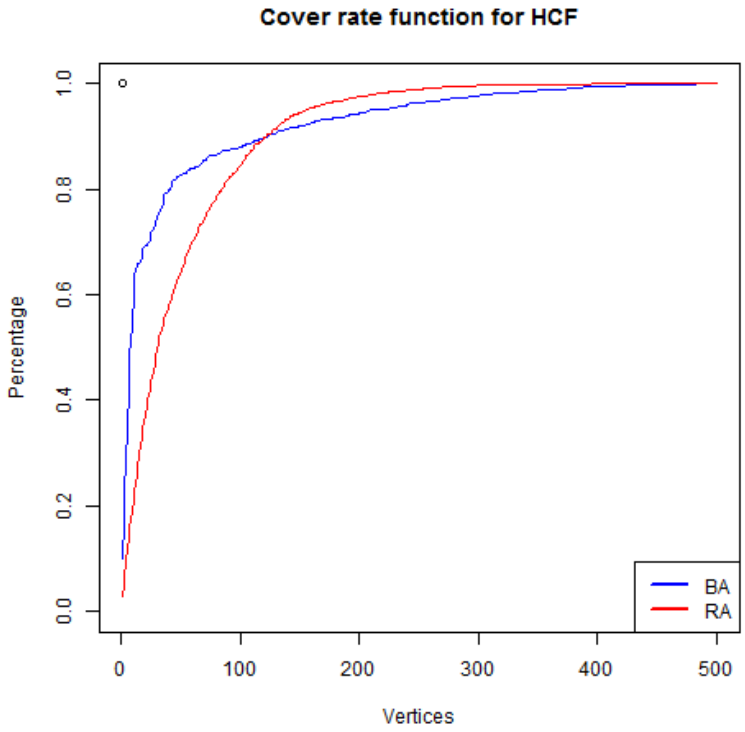
Betweenness centrality



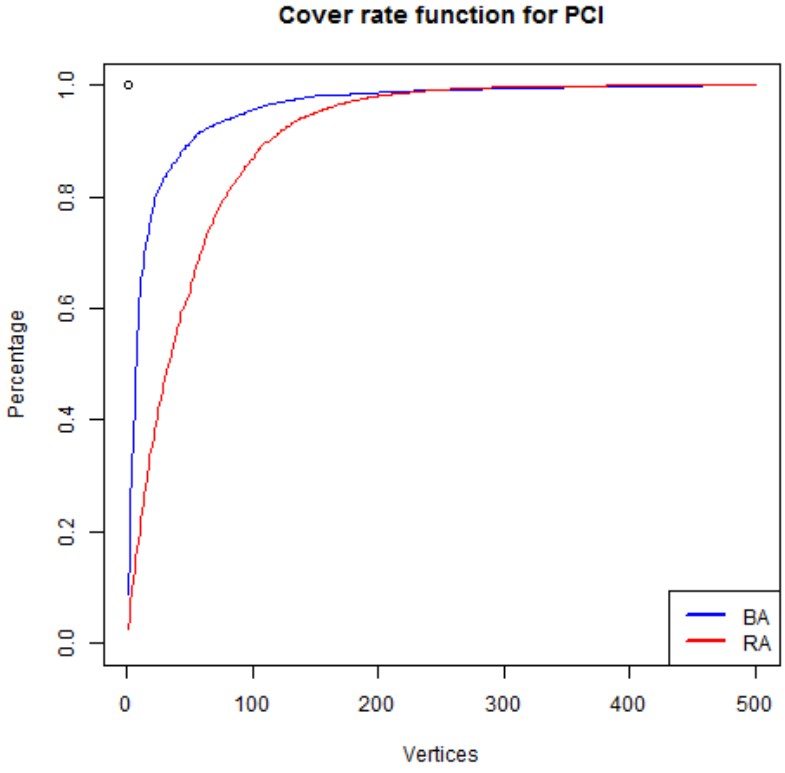
Degree Centrality



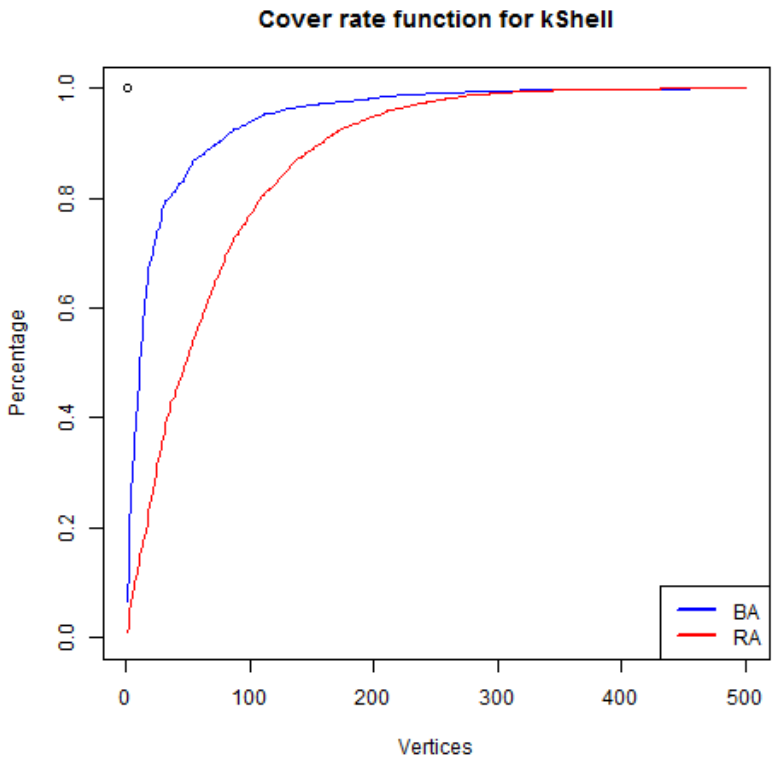
Closeness Centrality



PCI



kShell

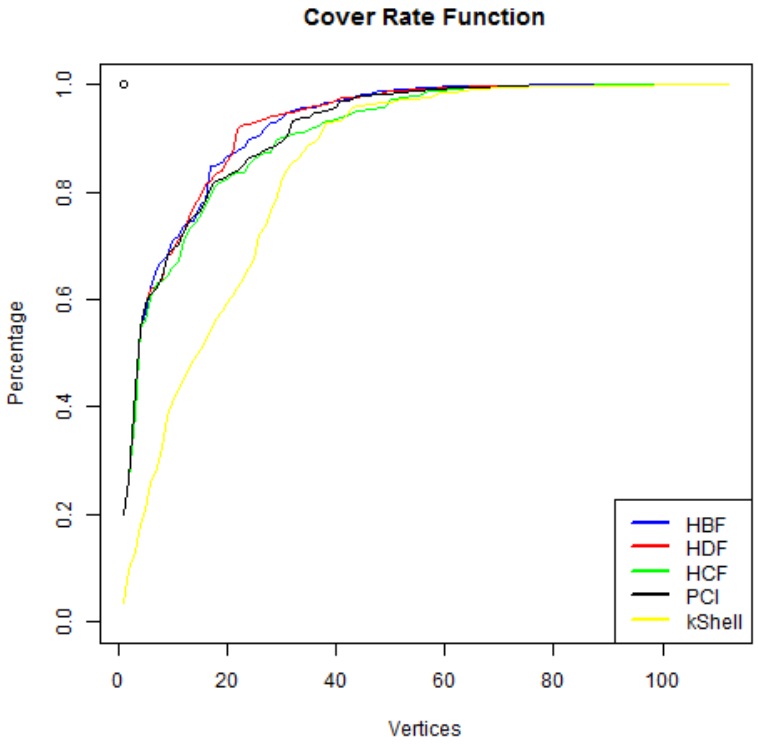


Και στα κατευθυνόμενα και στα μη κατευθυνόμενα όλες οι κεντρικότητες έχουν καλύτερα αποτελέσματα για scale free δίκτυα. Χρειάζονται λιγότερους κόμβους για να καλύψουν ολόκληρο το δίκτυο, αλλά έχουν και πιο απότομες γραφικές παραστάσεις (cover rate functions) για τους πρώτους ταξινομημένους κόμβους.

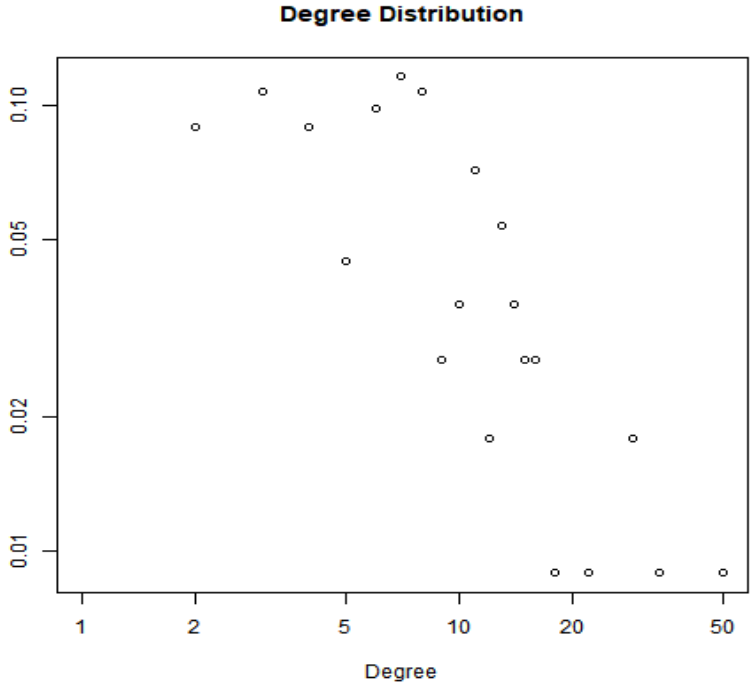
Κεφάλαιο 5

Εφαρμογή αλγορίθμου σε πραγματικά δίκτυα

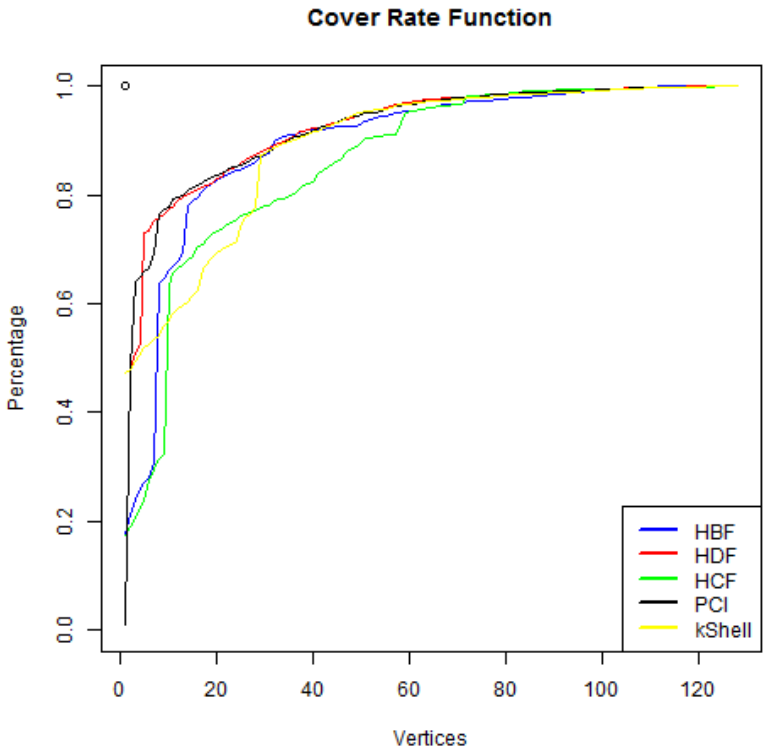
1.	Number of nodes	Number of edges	Average Degree	HBF	HDF	HCF	PCI	KSHELL
David Copperfield	112	425	7.5893	**76**	90	88	99	99
Florida Ecosystem Dry	128	2137	33.391	**112**	121	126	122	122
PDZBase	212	244	2.3019	**210**	**210**	211	**210**	**210**
Little Rock Lake	183	2494	27.257	180	145	**138**	147	147
Euroroad	1174	1417	2.4140	1174	**1173**	**1173**	**1173**	**1173**
Email	1133	5451	9.6222	1090	**979**	1084	1103	1103
Protein	1870	2277	2.4353	1867	1869	**1845**	1869	1869
Hamsterster friendship	1858	12534	13.492	**1820**	**1820**	1857	1830	1830
Human protein (Figeys)	2239	6452	5.7633	2066	**1778**	2238	1984	1983
Hamsterster full	2426	16631	13.711	**2224**	**2224**	2225	**2224**	**2224**
Human Protein(Vidal)	3133	3626	4.2936	3130	3130	**3022**	3130	3130



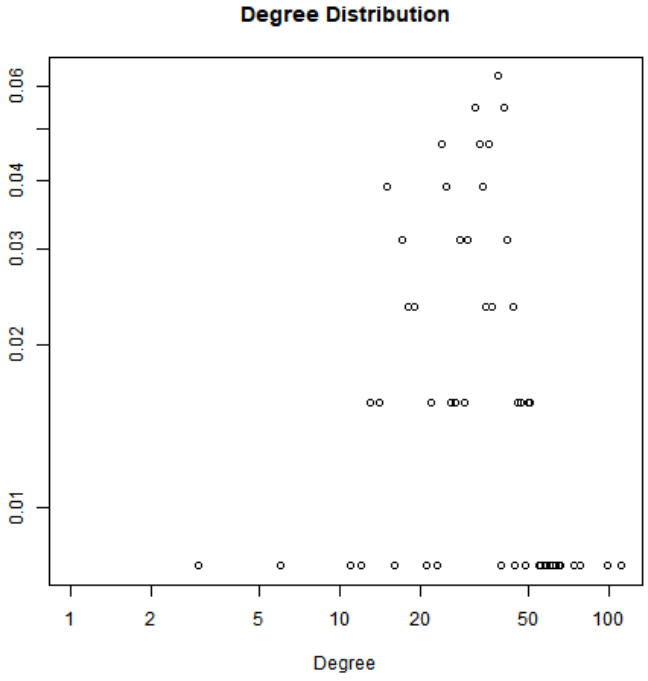
Centrality	Number of check in nodes	Percentage
Betweenness	76	76/112 = 67%
Degree	90	90/112 = 80 %
Closeness	88	88/112 = 78 %
PCI	99	99/112 =88%
kShell	99	99/112 = 88%



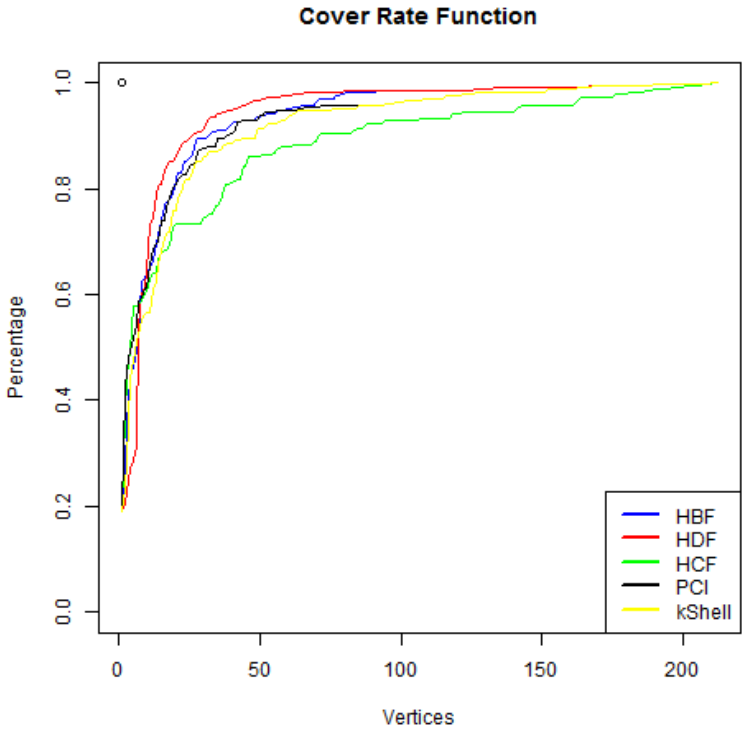
Florida ecosystem dry



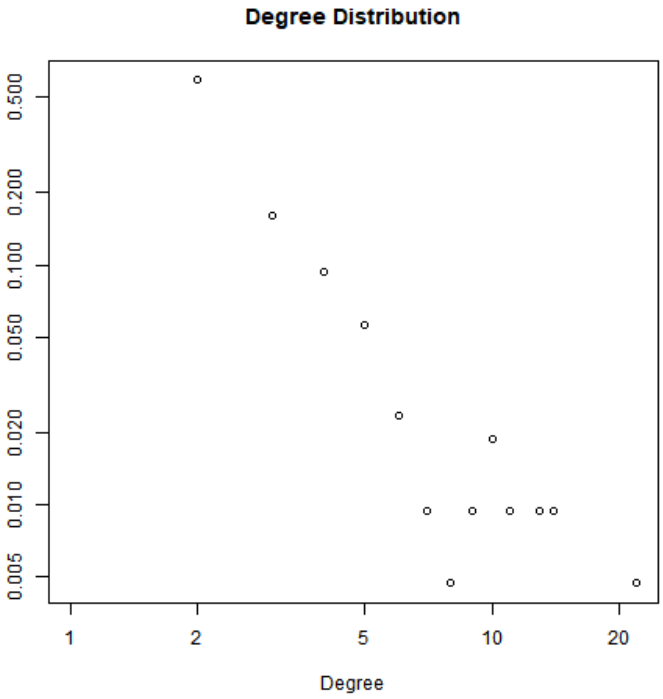
Centrality	Number of check in nodes	Percentage
Betweenness	112	112/128 = 87,5%
Degree	121	121/128 = 94,5 %
Closeness	126	126/128 = 98,4%
PCI	122	122/128 = 95,3 %
kShell	122	122/128 = 95,3 %



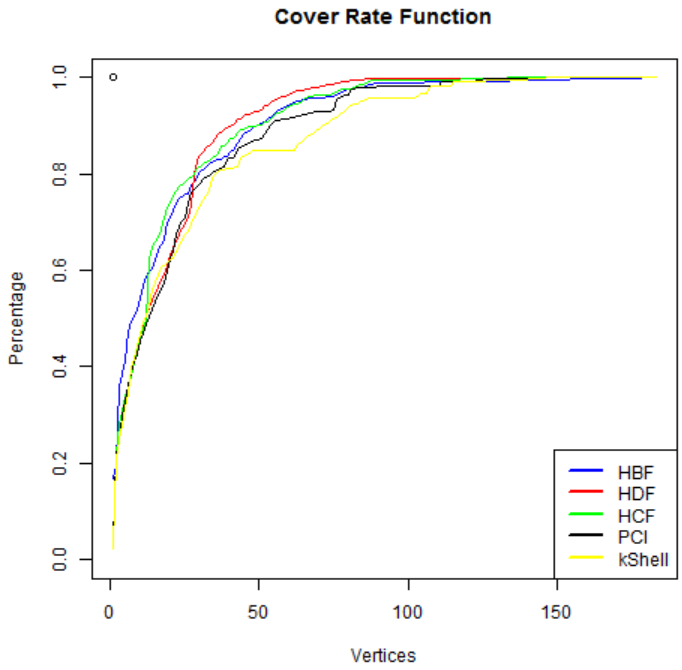
PDZBase



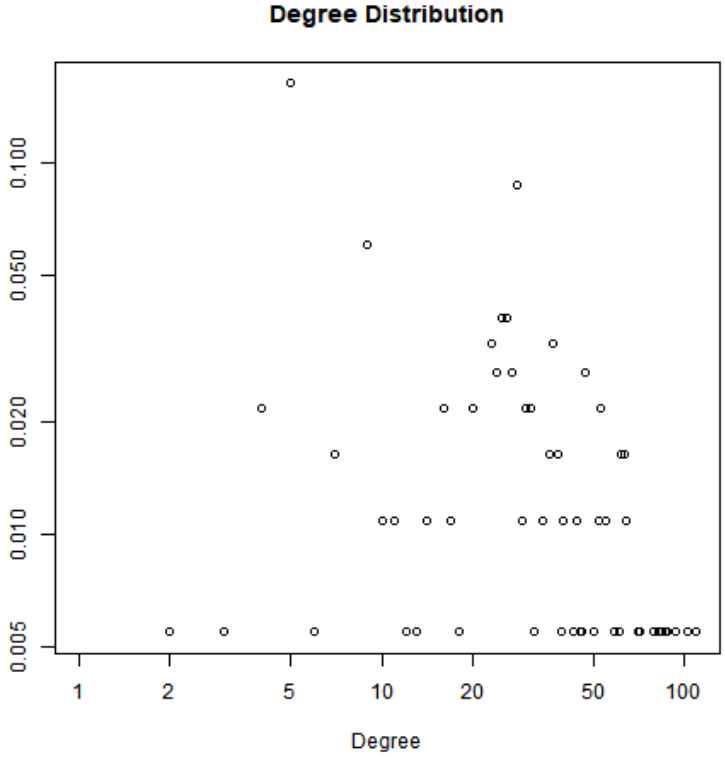
Centrality	Number of check in nodes	Percentage
Betweenness	210	210/212 = 99,05 %
Degree	210	210/212 = 99,05 %
Closeness	211	211/212 = 99,5 %
PCI	210	210/212 = 99,05 %
kShell	210	210/212 = 99,05 %



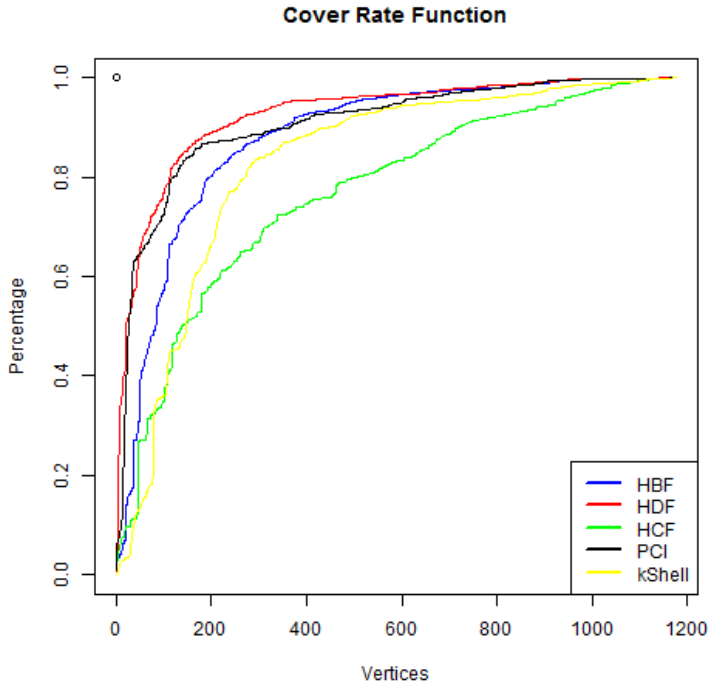
Little rock lake



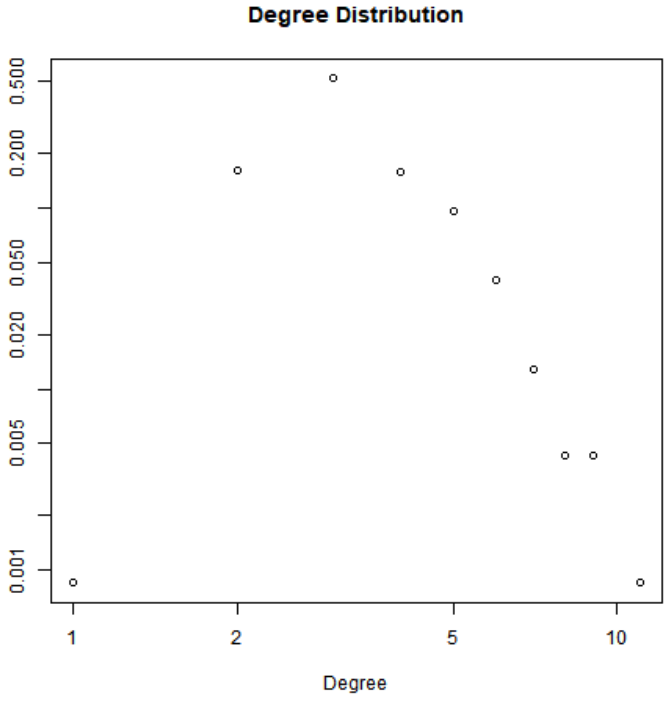
Centrality	Number of check in nodes	Percentage
Betweenness	180	180/183 = 98,3 %
Degree	145	145/183 = 79,2 %
Closeness	138	138/183 = 75,4 %
PCI	147	147/183 = 80,3 %
kShell	147	147/183 = 80,3 %



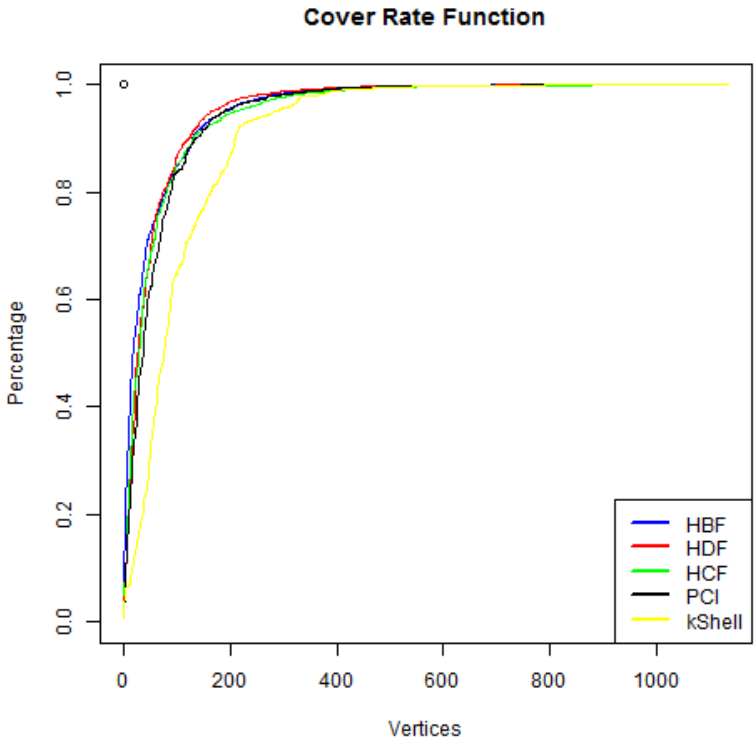
Euroroad



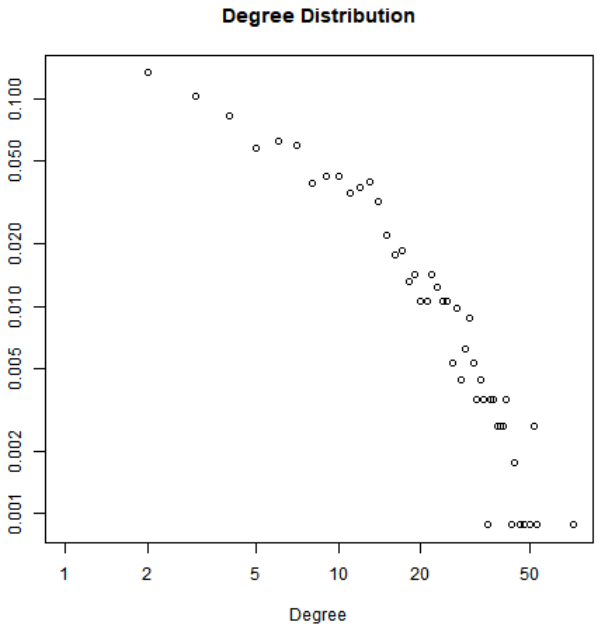
Centrality	Number of check in nodes	Percentage
Betweenness	1174	1174/1174 = 100%
Degree	1173	1173/1174 = 99,9 %
Closeness	1173	1173/1174 = 99,9 %
PCI	1173	1173/1174 = 99,9 %
kShell	1173	1173/1174 = 99,9 %



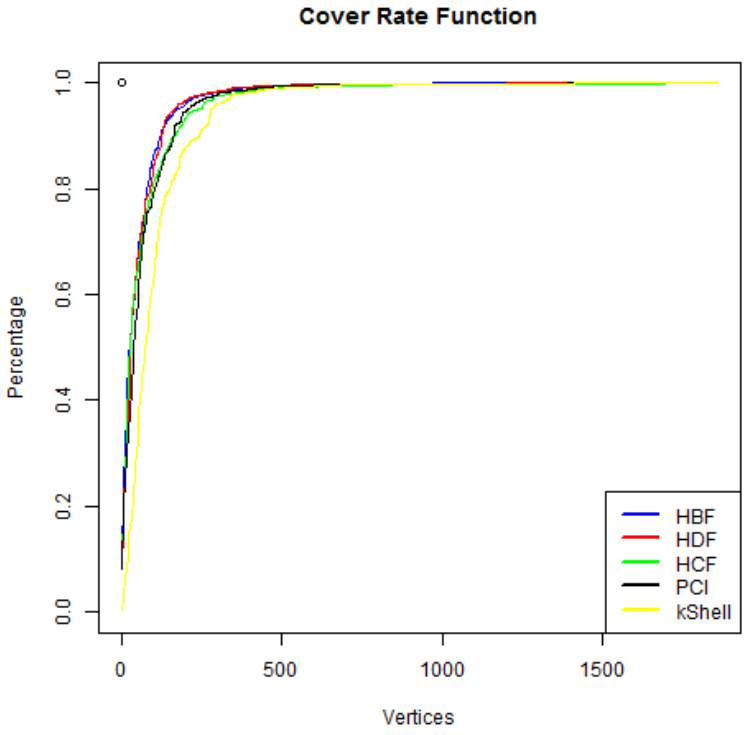
Email



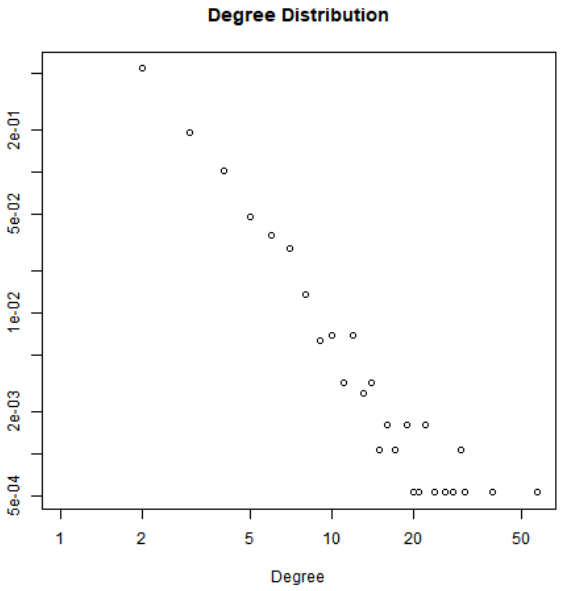
Centrality	Number of check in nodes	Percentage
Betweenness	1090	1090/1133 = 96,2 %
Degree	979	979/1133 = 86,4%
Closeness	1084	1084/1133 = 95,6 %
PCI	1103	1103/1133 = 97,3 %$
kShell	1103	1103/1133 = 97,3 %$



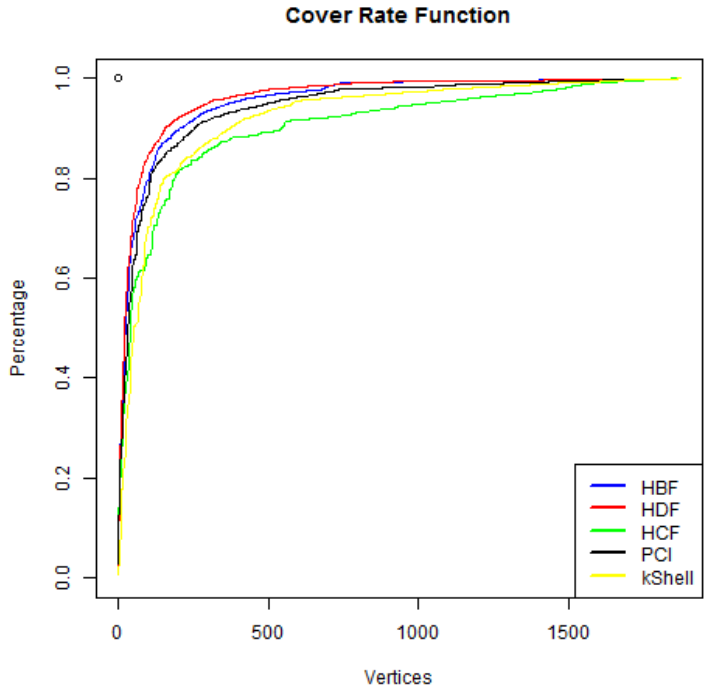
Protein



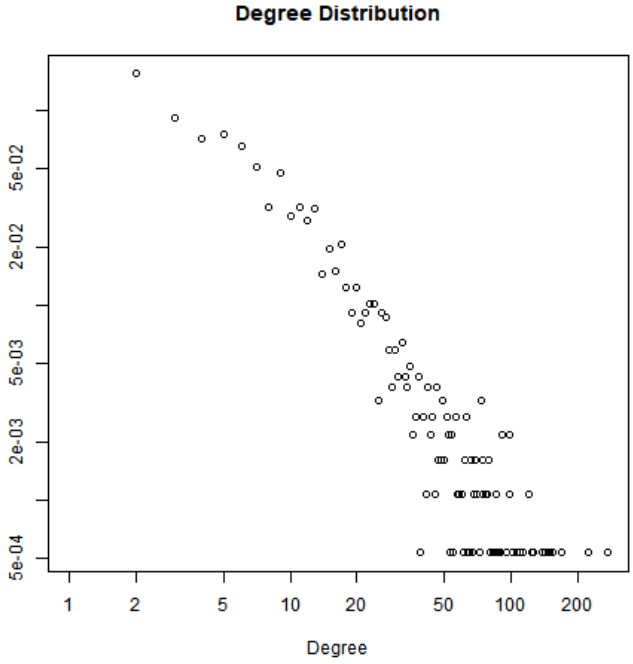
Centrality	Number of check in nodes	Percentage
Betweenness	1867	1867/1870 = 99,8 %
Degree	1869	1869/1870 = 99,9 %
Closeness	1845	1845/1870 = 98,6 %
PCI	1869	1869/1870 = 99,9 %
kShell	1869	1869/1870 = 99,9 %



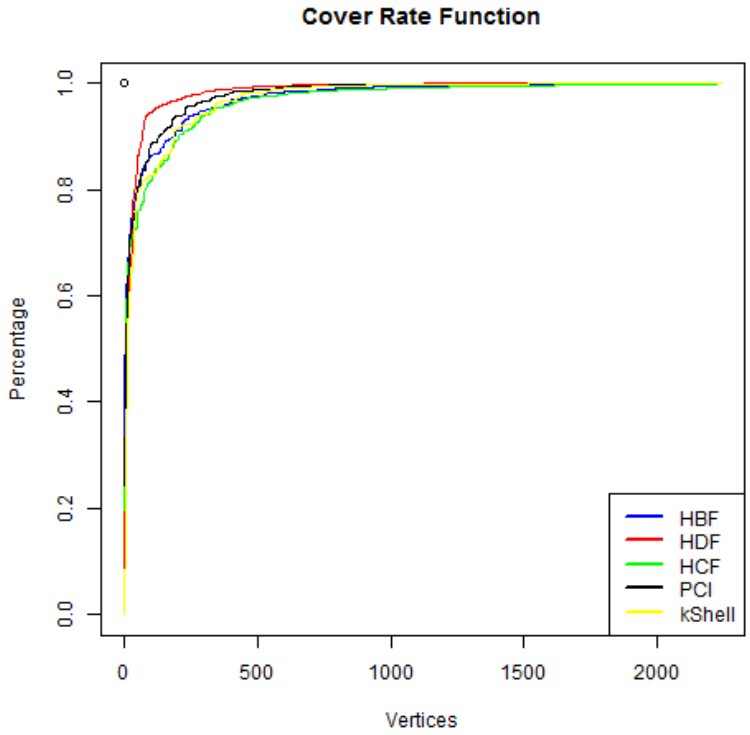
Hamsterster friendship



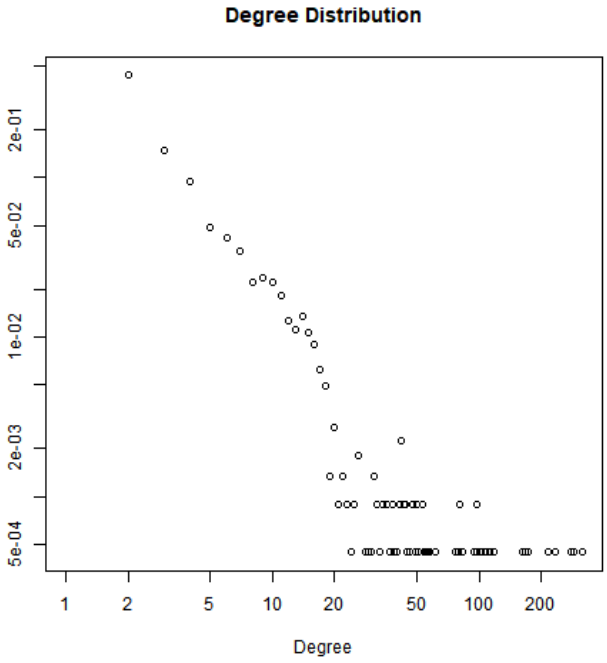
Centrality	Number of check in nodes	Percentage
Betweenness	1820	1820/1858 = 97,9 %
Degree	1820	1820/1858 = 97,9 %
Closeness	1857	1857/1857 = 99,9 %
PCI	1830	1830/1857 = 98,5 %
kShell	1830	1830/1857 = 98,5 %



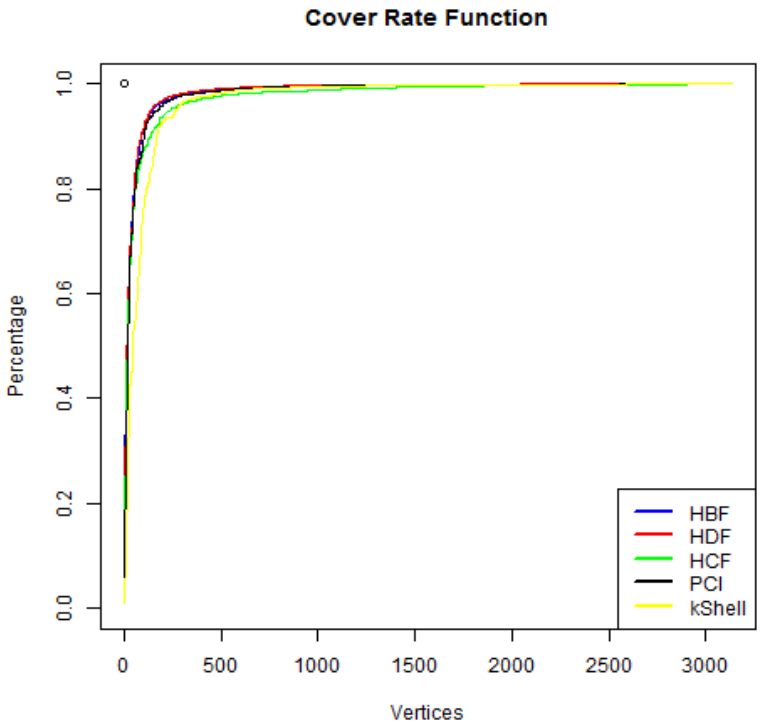
Human Protein (Figeys)



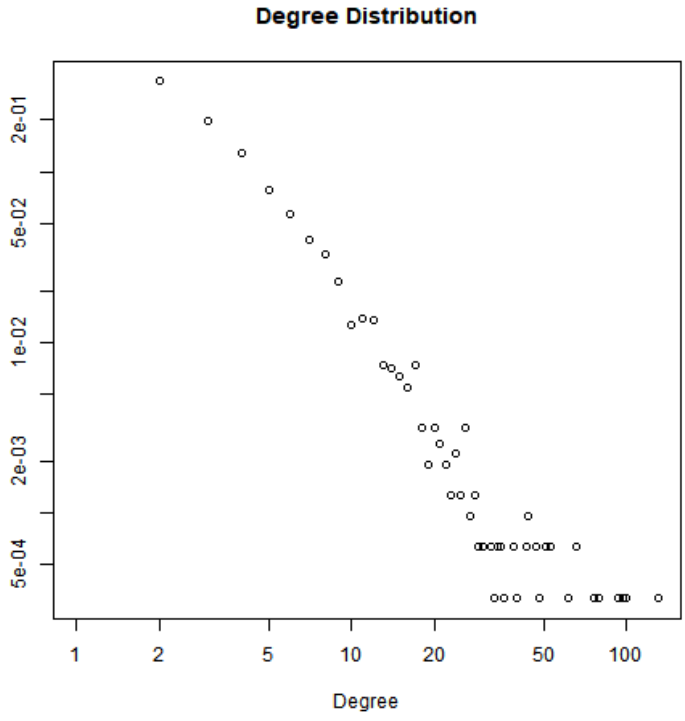
Centrality	Number of check in nodes	Percentage
Betweenness	2066	2066/2239 = 92,2 %
Degree	1778	1778/2239 = 79,4 %
Closeness	2238	2238/2239 = 99,9 %
PCI	1984	1984/2239 = 88,6 %
kShell	1983	1983/2239 = 88,5 %



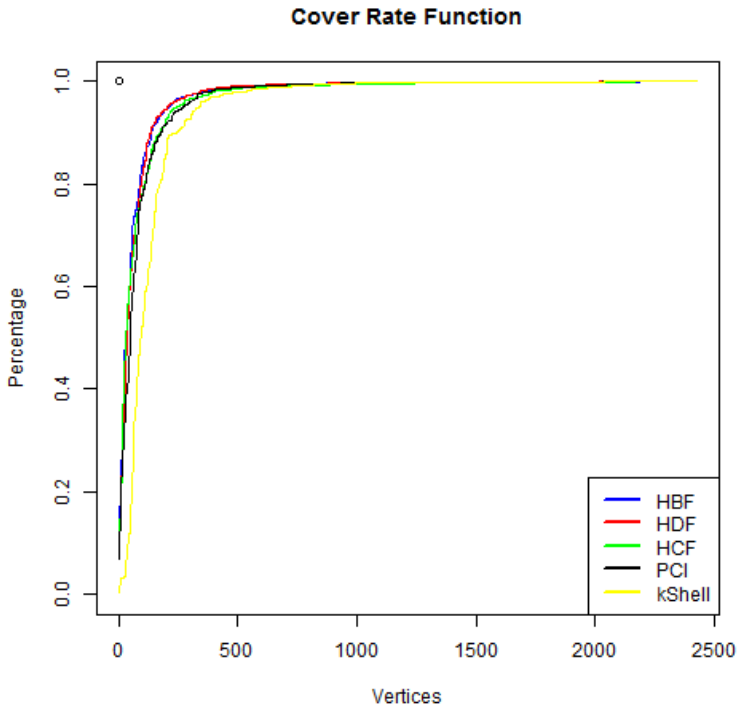
Human Protein(Vidal)



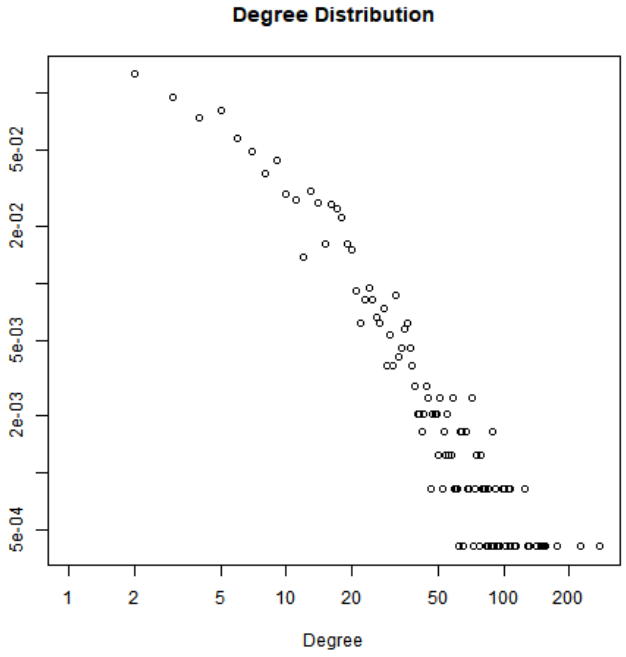
Centrality	Number of check in nodes	Percentage
Betweenness	3130	3130/3133 = 99,9 %
Degree	3130	3130/3133 = 99,9 %
Closeness	3022	3022/3133 = 96,4 %
PCI	3130	3130/3133 = 99,9 %
kShell	3130	3130/3133 = 99,9 %



Hamsterster full



Centrality	Number of check in nodes	Percentage
Betweenness	2224	2224/2426 = 91,6 %
Degree	2224	2224/2426 = 91,6 %
Closeness	2225	2225/2426 = 91,7 %
PCI	2224	2224/2426 = 91,6 %
kShell	2224	2224/2426 = 91,6 %



Τα παραπάνω πραγματικά δίκτυα χωρίζονται σε ζευγάρια με ίδιο περίπου αριθμό κόμβων αλλά (αρκετά) διαφορετικό πλήθος ακμών. Προσπάθησα τα δίκτυα να είναι διαφορετικά μεταξύ τους και αυτό φαίνεται στο average degree. Για παράδειγμα στο ζευγάρι δικτύων David Copperfield και Florida Ecosystem Dry, το πρώτο δίκτυο έχει μικρότερο average degree. Το ίδιο ισχύει και για τα υπόλοιπα ζευγάρια δικτύων.

Παρατηρούμε πως τα αποτελέσματα συγκλίνουν με τις προσομοιώσεις που κάναμε. Η betweenness centrality και η degree centrality είναι πιο πιθανό να μας δώσουν τους ελάχιστους κόμβους καταχώρησης. Υπάρχουν όμως και δίκτυα στα οποία η closeness centrality μας δίνει τους ελάχιστους κόμβους καταχώρησης όπως και δίκτυα όπου το PCI και το kshell μαζί με άλλες κεντρικότητες όπου μας δίνουν τους ελάχιστους κόμβους καταχώρησης. Επίσης κοιτώντας τις cover rate functions όλων των centralities παρατηρούμε πως είναι πολύ κοντά η μία με την άλλη, γεγονός που συνέβαινε και στις προσομοιώσεις.

Κεφάλαιο 6

Μελλοντική εργασία

Παραπάνω λύσαμε το πρόβλημα των ελάχιστων κόμβων καταχώρησης σε ένα δίκτυο χρησιμοποιώντας τις κεντρικότητες των σύνθετων δικτύων. Απαράβατη προϋπόθεση ήταν να καλύψουμε ολόκληρο το δίκτυο μας, οι κόμβοι καταχώρησης να καλύπτουν όλα τα συντομότερα μονοπάτια. Θα μπορούσαμε να προσθέσουμε λειτουργικότητα στον κώδικα μας προκειμένου να καλύψουμε τα παρακάτω σενάρια:

1. Έστω ένα δίκτυο με Ν κόμβους. Δεδομένου ότι μπορούμε να διαθέσουμε συγκεκριμένο αριθμό κόμβων καταχώρησης (π.χ. Ν/10), πόσο μέρος του δικτύου καλύπτω; Ποια κεντρικότητα έχει τα καλύτερα αποτελέσματα; Ποιο μέρος του δικτύου είναι ακάλυπτο;
2. Επίλυση του προβλήματος των ελάχιστων κόμβων καταχώρησης και για διαφορετικές στρατηγικές δρομολόγησης. Σε αυτή την εργασία ως στρατηγική δρομολόγησης χρησιμοποιήθηκε η μέθοδος των συντομότερων μονοπατιών. Υπάρχουν και άλλες στρατηγικές δρομολόγησης όπως η αποδοτική δρομολόγηση, η δρομολόγηση σε τοπικό επίπεδο, η δρομολόγηση σε παγκόσμιο επίπεδο κ.α.
3. Σε αυτή την εργασία, αν μεταξύ δυο κόμβων υπάρχουν παραπάνω από ένα συντομότερα μονοπάτια, διαλέγουμε ένα στη τύχη. Άμα θέλουμε να εμβαθύνουμε ακόμα παραπάνω, μπορούμε με βάση κάποιο κριτήριο να διαλέγουμε ένα συντομότερο μονοπάτι. Ίσως άμα θέτουμε ένα άνω όριο στον αριθμό των συντομότερων μονοπατιών που μπορεί να εξυπηρετήσει ένας κόμβος. Το παραπάνω σενάριο ίσως έχει πρακτική εφαρμογή σε δίκτυα ηλεκτροδότησης και τηλεπικοινωνιών.
4. Σε αυτή την εργασία χρησιμοποιήθηκαν πέντε κεντρικότητες, οι degree, betweenness, closeness, PCI και kShell centralities. Υπάρχουν και άλλες κεντρικότητες που ίσως έχουν καλύτερα αποτελέσματα.
5. Στην ταξινόμηση των κόμβων με βάση τις κεντρικότητες PCI και kShell, υπάρχουν πολλοί κόμβοι με ίδιο PCI ή kShell. Ίσως άμα ταξινομήσουμε αυτούς τους κόμβους με κάποιο κριτήριο (π.χ. με βάση το degree τους) να έχουμε καλύτερα αποτελέσματα.
6. Κάποια πραγματικά δίκτυα, ίσως είναι χωρισμένα σε υποδίκτυα. Μπορούμε να προσθέσουμε λειτουργικότητα έτσι ώστε να αναγνωρίζουμε τους κόμβους καταχώρησης για κάθε ένα υποδίκτυο.

Βιβλιογραφία/Αναφορές

https://cran.r-project.org/

http://igraph.org/redirect.html

http://inf-server.inf.uth.gr/courses/CE427

http://konect.uni-koblenz.de/networks/adjnoun_adjacency

http://konect.uni-koblenz.de/networks/foodweb-baydry

http://konect.uni-koblenz.de/networks/maayan-pdzbase

http://konect.uni-koblenz.de/networks/maayan-foodweb

http://konect.uni-koblenz.de/networks/subelj_euroroad

http://konect.uni-koblenz.de/networks/arenas-email

http://konect.uni-koblenz.de/networks/moreno_propro

http://konect.uni-koblenz.de/networks/petster-friendships-hamster

http://konect.uni-koblenz.de/networks/maayan-figeys

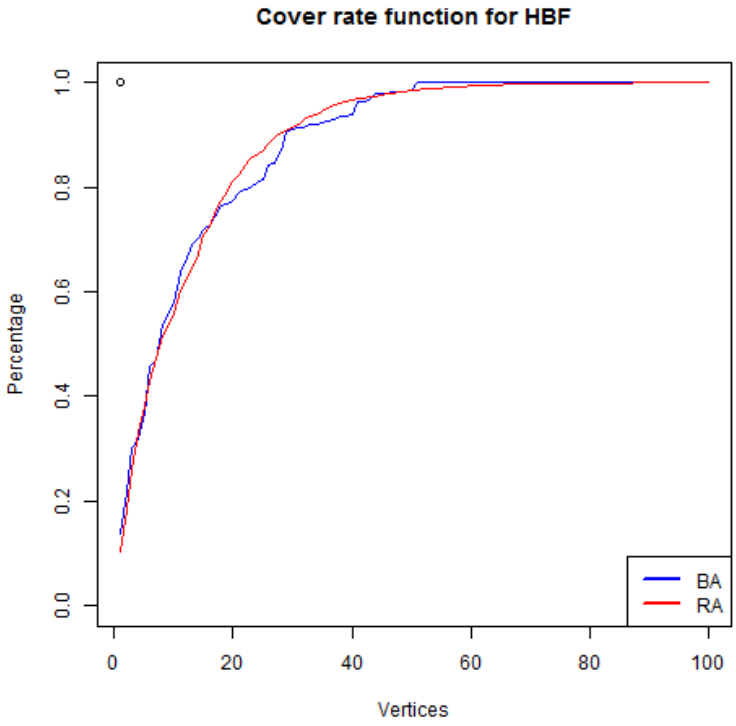
http://konect.uni-koblenz.de/networks/petster-hamster

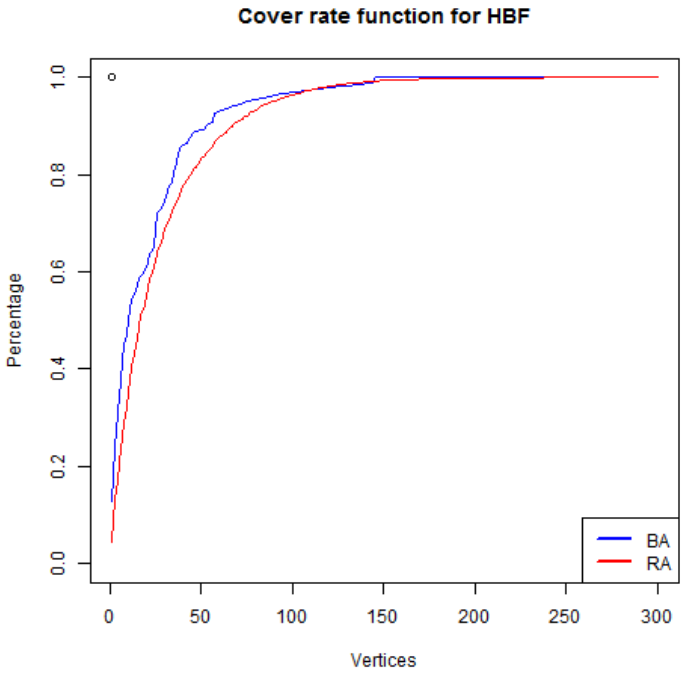
http://konect.uni-koblenz.de/networks/maayan-vidal

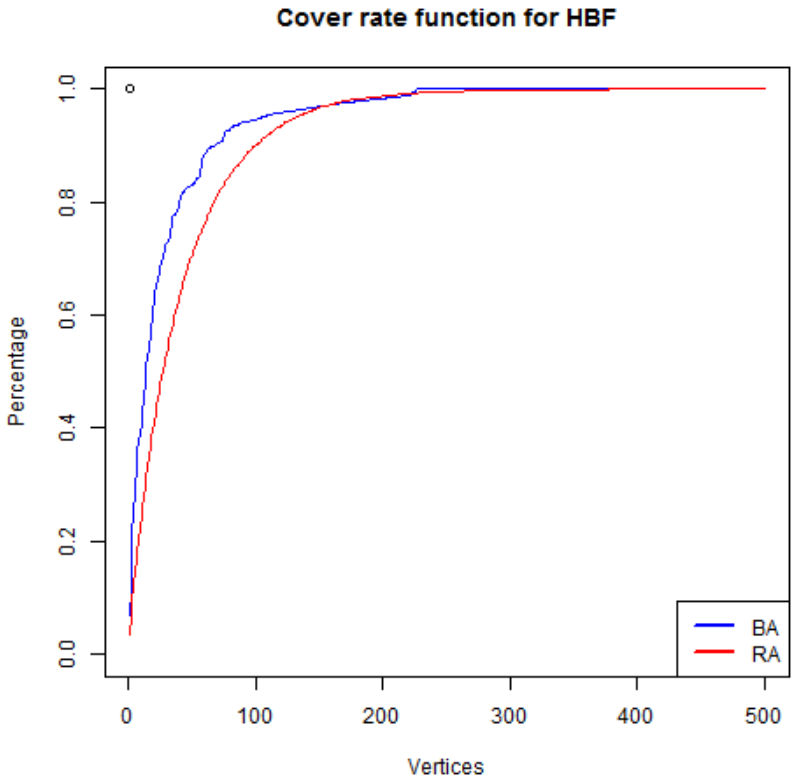
http://konect.uni-koblenz.de/networks/opsahl-openflights

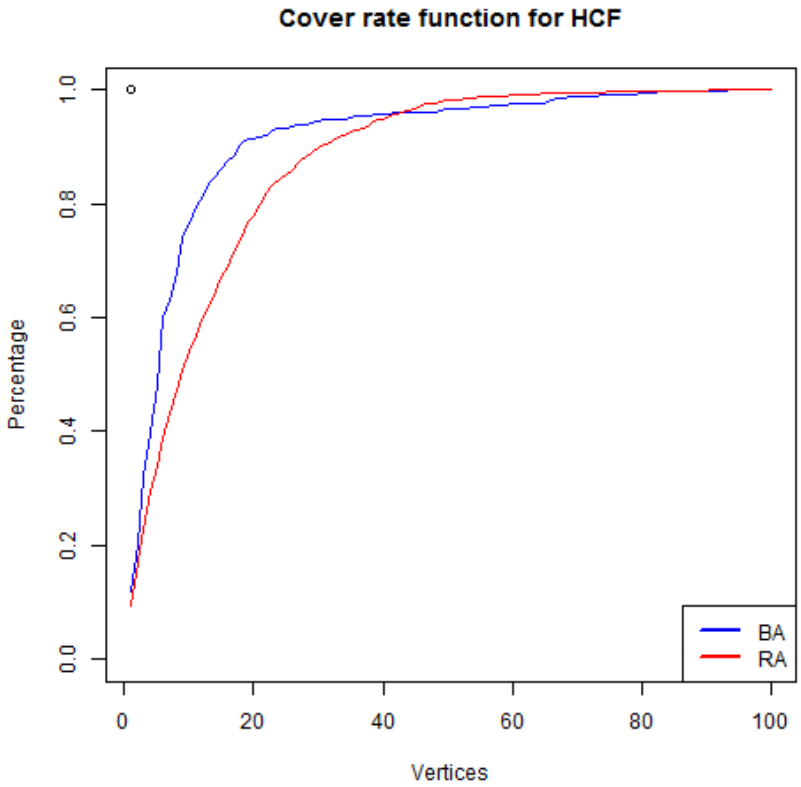
Παράρτημα Α

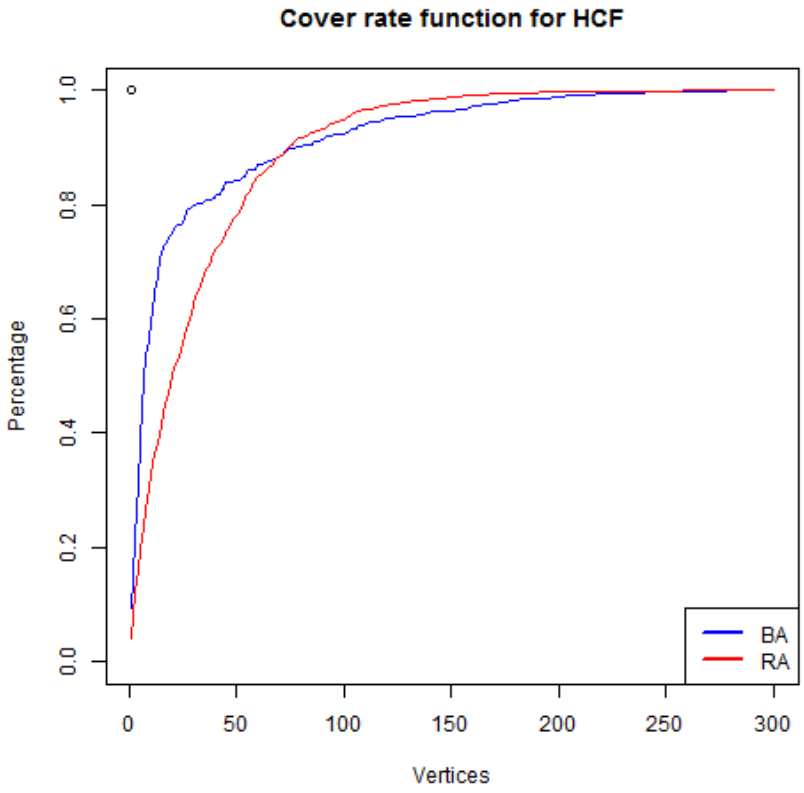
Κατευθυνόμενα Δίκτυα

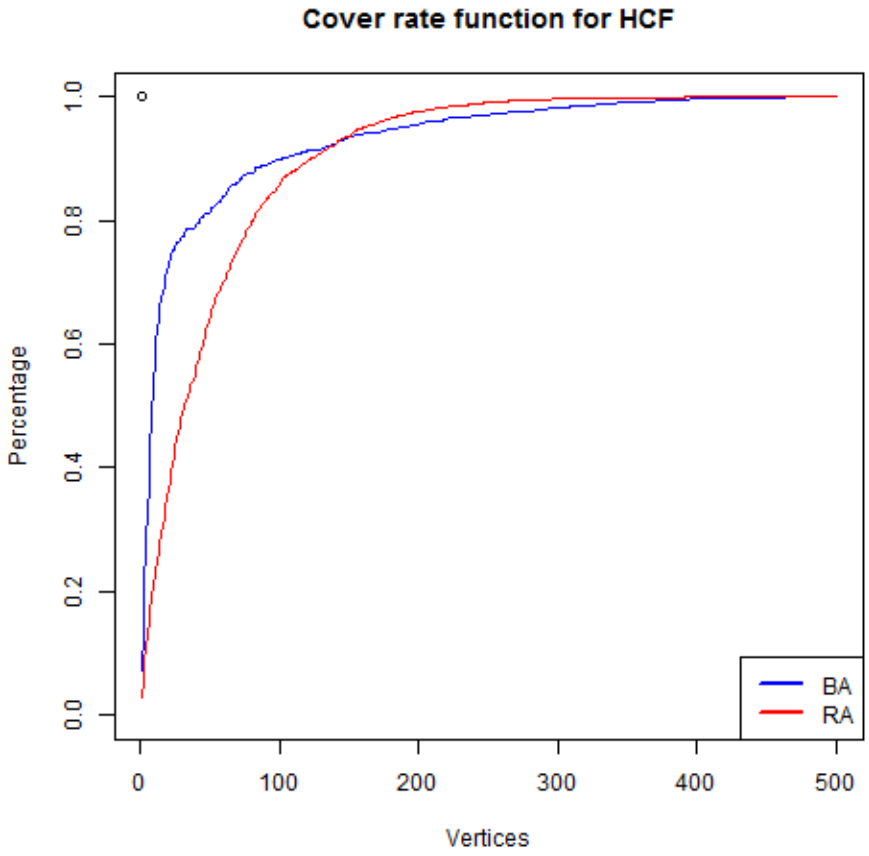


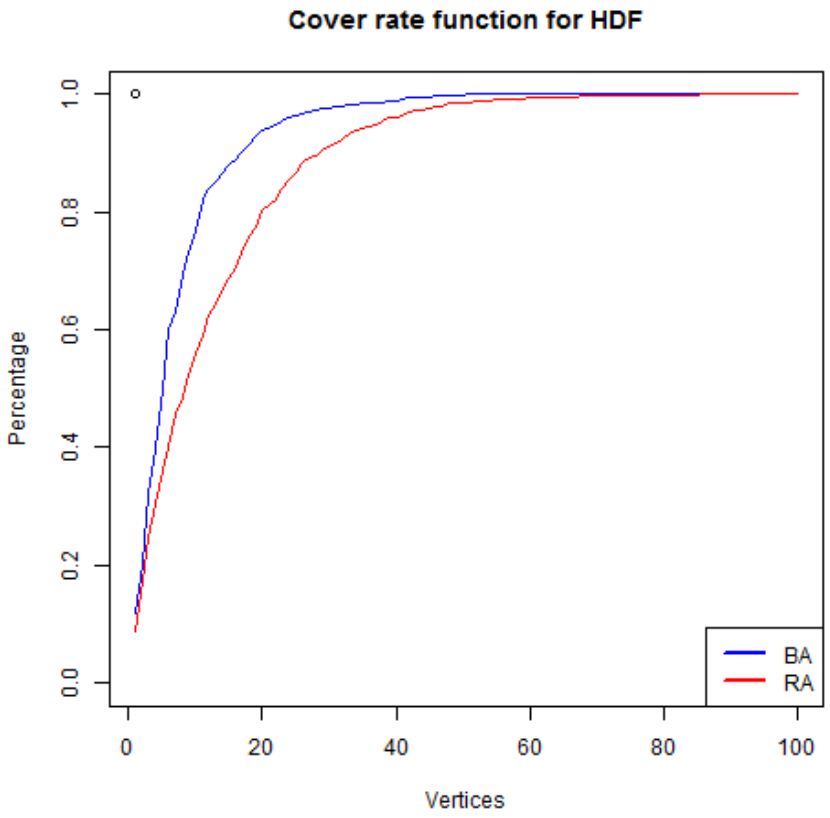


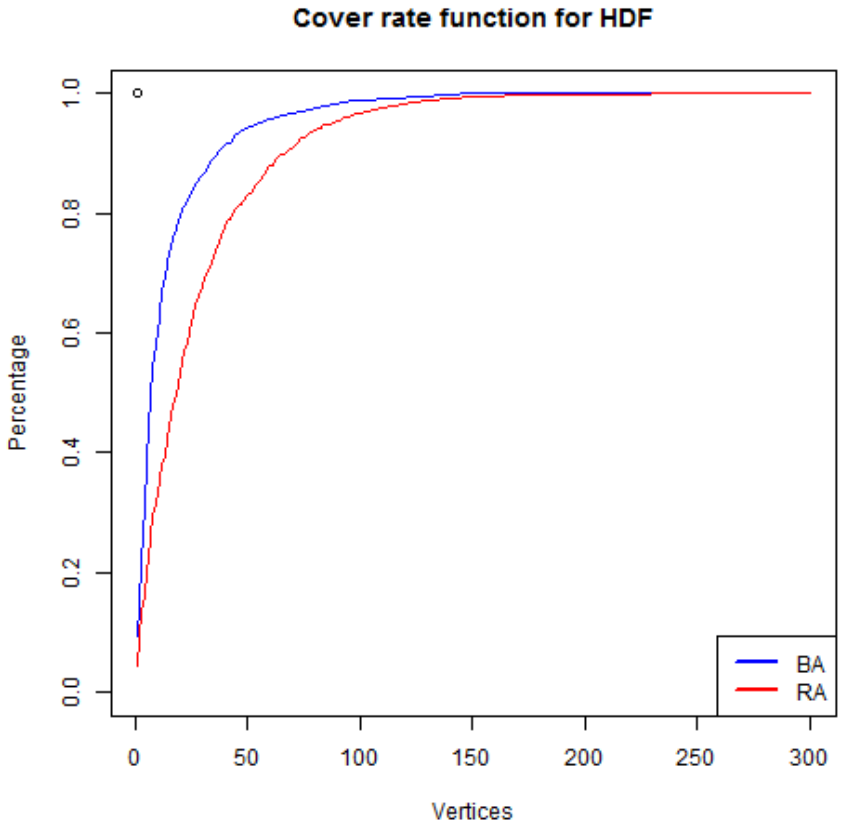


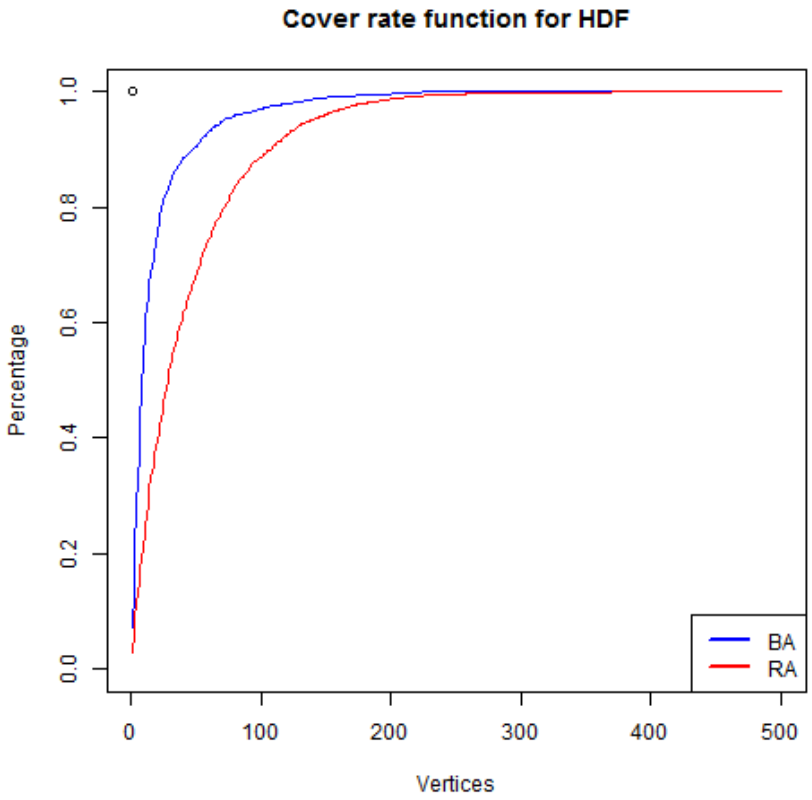


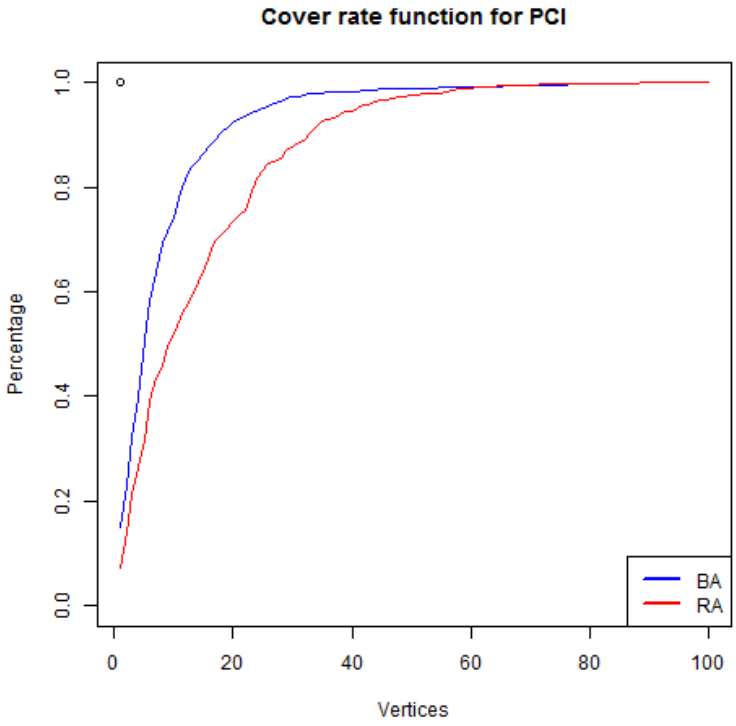


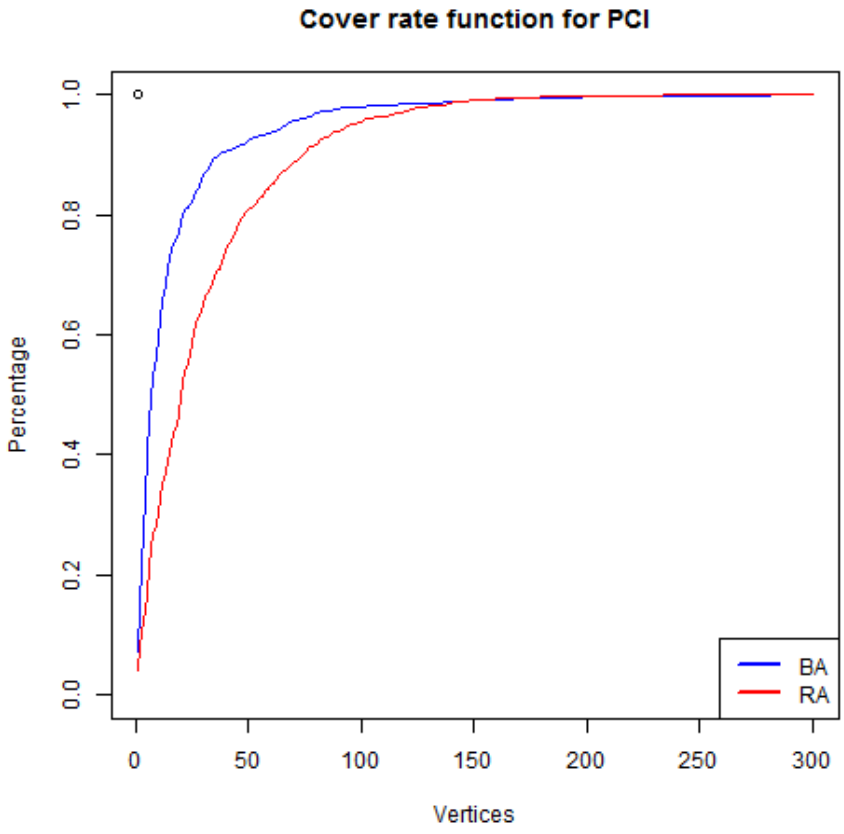


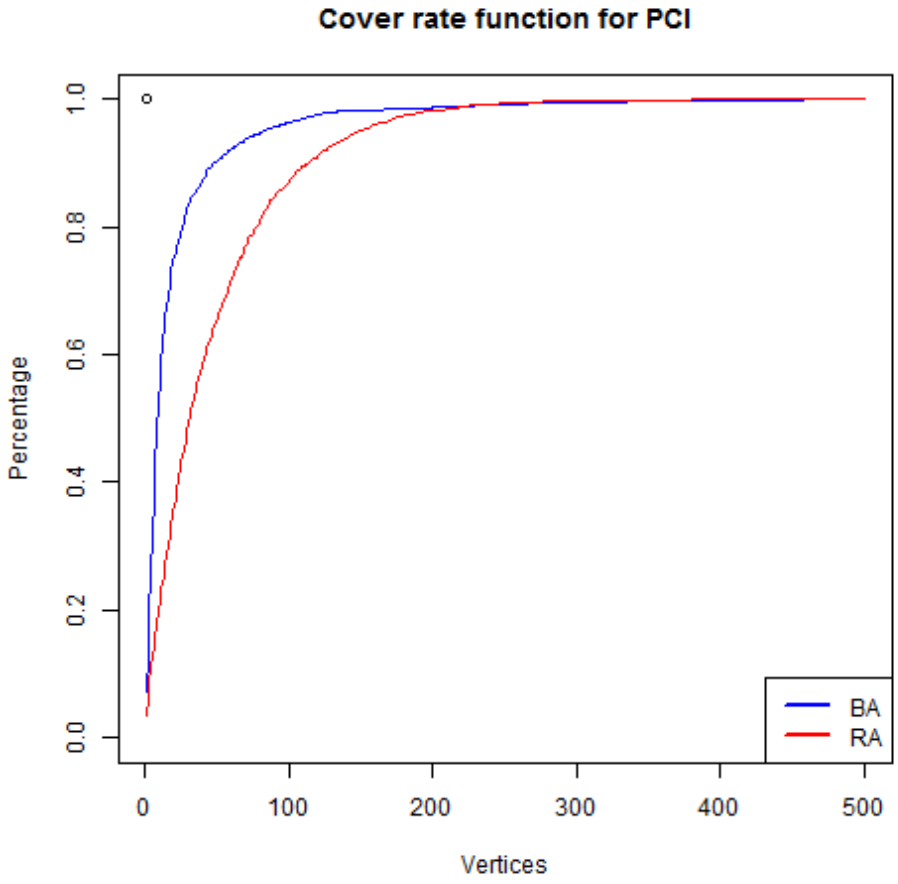


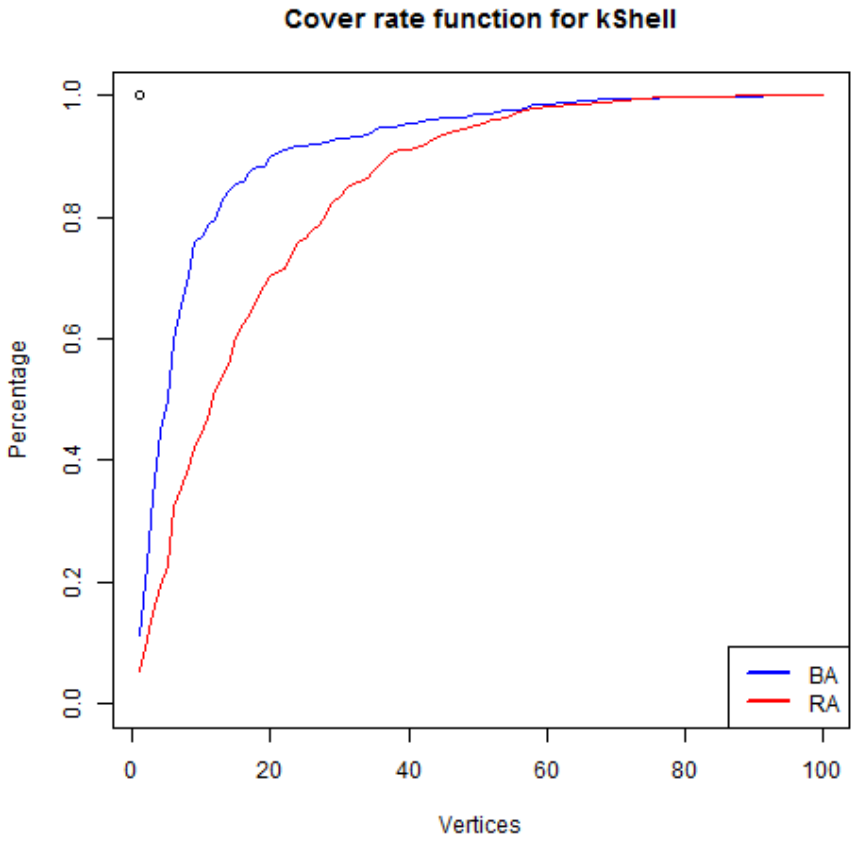


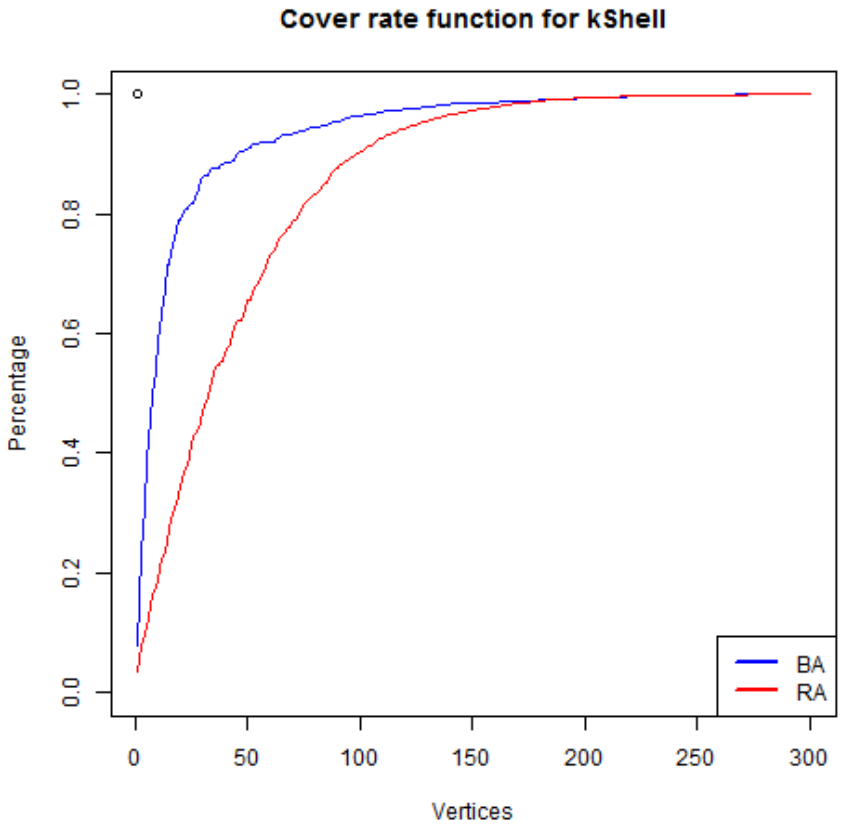


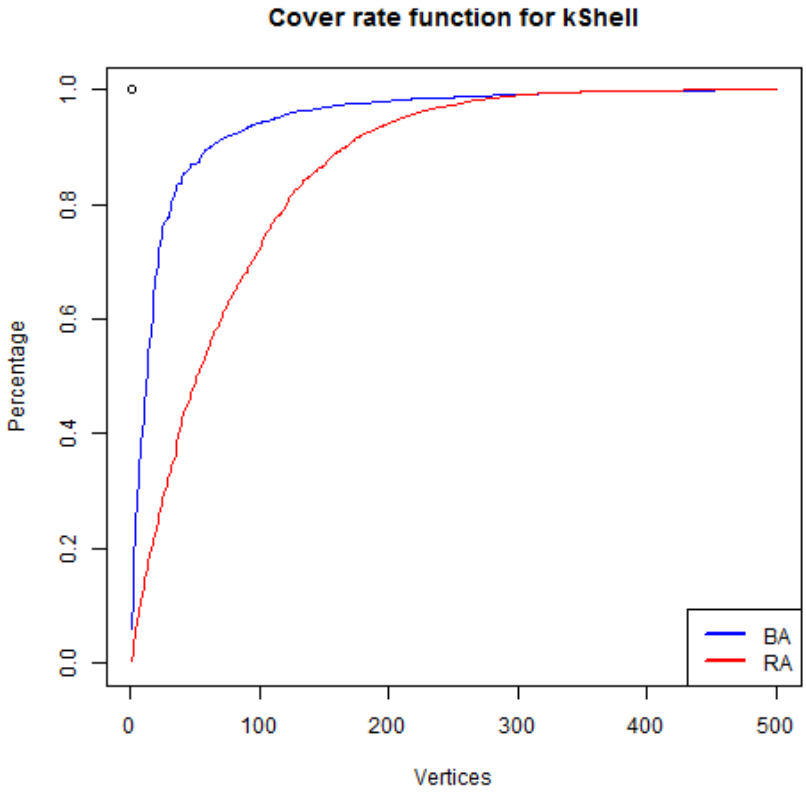












Μη κατευθυνόμενα Δίκτυα